

ПРОБЛЕМЫ ИНФОРМАТИКИ



ТЕОРЕТИЧЕСКАЯ И СИСТЕМНАЯ ИНФОРМАТИКА
ПРИКЛАДНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ
ПАРАЛЛЕЛЬНОЕ СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ
И ВЫЧИСЛИТЕЛЬНЫЕ ТЕХНОЛОГИИ

1
—
2021

ПРОБЛЕМЫ ИНФОРМАТИКИ № 1 (50) 2021 г.

Журнал выходит ежеквартально, издается с 2008 г.

Учредитель журнала — Институт вычислительной математики и математической геофизики СО РАН при поддержке Института информационных и вычислительных технологий МОН РК.

Редакционный совет

Председатель — акад. НАН РК М. Н. Калимолдаев,

Акад. РАН А. Л. Асеев, акад. АН РУ Т. Ф. Бекмуратов (Республика Узбекистан), проф.

В. А. Васенин, акад. РАН С. Н. Васильев, проф. В. М. Вишневский, акад. РАН С. С. Гончаров,

акад. РАН Ю. Л. Ершов, акад. РАО А. А. Кузнецов, акад. РАН Н. А. Кузнецов, акад. РАН

А. П. Кулешов, проф. А. Г. Марчук, проф. Б. Я. Рябко, проф. Н. А. Семенов, акад. РАН

И. А. Соколов, проф. А. Н. Сотников, чл.-кор. РАН Ю. А. Флеров, проф. П. С. Чубик, акад.

НАН КР Ж. Ш. Шаршеналиев (Кыргызская Республика).

Редколлегия

Главный редактор — проф. В. Э. Малышкин,

Д. Ж. Ахмед-Заки, А. Г. Вострецов, В. П. Гергель, Б. С. Гольдштейн, В. И. Гужов,

Ю. А. Загорулько, С. Д. Каракозов, М. М. Каримов, В. Н. Касьянов, О. В. Кибис, В. В. Корнеев,

И. В. Котенко, Т. П. Любимова, А. И. Ляхов, М. А. Марченко, В. В. Окольнишников,

Б. В. Поллер, А. С. Родионов (зам. гл. редактора), М. А. Сонькин, В. В. Шахов (зам. гл.

редактора), М. С. Хайретдинов, Ph. D. Moonseong Kim (Korea), Prof. Dr.-Eng. V. D. Nguyen (Vietnam), Michele Pagano (Italy).

Редакция: отв. секретарь М. С. Делидович, системный администратор В. А. Перепелкин, верстка Д. В. Лазуткин, логист Л. В. Трофимова.

Адрес редакции, издателя: 630090, г. Новосибирск, просп. Академика Лаврентьева, д. 6, ИВМ и МГ СО РАН

тел. (383) 330-96-43; e-mail: problem-info@sscc.ru, <http://www.problem-info.sscc.ru>.

Журнал зарегистрирован в Роскомнадзоре. Свидетельство ПИ № ФС77-32088 от 27 мая 2008 г.

Подписной индекс в каталоге „Издания органов научно-технической информации“

ОАО «Агентство „Роспечать“» — 69980. Цена свободная. Журнал распространяется на территории России.

Журнал включен в Перечень ведущих рецензируемых научных журналов, рекомендованных для публикаций Высшей аттестационной комиссией.

Все права авторов сохранены. Использование материалов журнала возможно только с разрешения редакции и авторов.

Отпечатано в типографии ТОО „Инфо-Алдин“. Адрес: 050009, Республика Казахстан, г. Алма-Ата, ул. Толе би, д. 188;

тел./факс 8 (727) 272-78-26. Формат 60 × 84 1/8. Усл. печ. л. _____. Уч.-изд. л. _____. Печать офсетная. Тираж 300 экз. Заказ № _____. Подписано в печать _____ г. Выход в свет _____ г.

© Институт вычислительной математики и математической геофизики СО РАН, 2021

© Институт информационных и вычислительных технологий МОН РК, 2021

JOURNAL “PROBLEMS OF INFORMATICS”. No. 1 (50) 2021

Publisher: Institute of Computational Mathematics and Mathematical Geophysics of Siberian Branch of Russian Academy of Sciences with the support of the Institute of Information and Computing Technologies of the Ministry of Education of the Republic of Kazakhstan.

Editorial Council

Chairman Academician of the National Academy of Sciences of the Republic of Kazakhstan
M. N. Kalimoldayev

Full Member of the RAS A. L. Aseev, Full Member of the Academy of Sciences of Uzbekistan
T. F. Bekmuratov, Professor V. A. Vasenin, Full Member of RAS C. N. Vassilyev, Professor
V. M. Vishnevsky, Full Member of RAS S. S. Goncharov, Full Member of RAS Yu. L. Ershov, Corr.
Member of RAE A. A. Kuznetsov, Full Member of RAS N. A. Kuznetsov, Full Member of RAS
A. P. Kuleshov, Professor A. G. Marchuk, Professor B. Y. Ryabko, Professor N. A. Semenov, Full
Member of RAS I. A. Sokolov, Professor A. N. Sotnikov, Corr. Member RAS Y. A. Flerov, Full
Member of NAS KR J. Sh. Sharshenaliev.

Editorial board

The Editor-in-Chief Professor V. E. Malyshkin

Associate Editors-in-Chief: A. S. Rodionov, V. V. Shakhov

D. Zh. Akhmed-Zaki, A. G. Vostretsov, V. P. Gergel, B. S. Goldstein, V. I. Guzhov, Y. A. Zagorulko,
S. D. Karakozov, M. M. Karimov, V. N. Kasyanov, O. V. Kibis, V. V. Korneev, I. V. Kotenko, T. P.
Lyubimova, A. I. Lyakhov, M. A. Marchenko, V. V. Okolnishnikov, B. V. Poller, Y. G. Soloveichik,
M. A. Sonkin, M. S. Khairetdinov, Moonseong Kim (Korea), Van Duc Nguyen (Vietnam), Michele
Pagano (Italy).

Editorial staff: Managing Editor M. S. Delidovich, System Administrator V. A. Perepelkin,
Maker-up D. V. Lazutkin, Logistician L. V. Trofimova.

Address of the editorial office: 630090, pr. Lavrentieva, 6, Novosibirsk, Russia, Institute of
Computational Mathematics and Mathematical Geophysics of SB RAS.

Phone: +7 (383) 330-96-43; e-mail: problem-info@sscc.ru, <http://www.problem-info.sscc.ru>.

The journal has been registered in accordance with Legislation of the Russian Federation. Certificate
of Mass Media Registration: ПИ № ФС77-32088, of 27 May, 2008, ISSN 2073-0667. The journal is
distributed in Russia.

The journal “Problems of Informatics” is in the List of Peer-Reviewed Scientific Journals for
publication of scientific results of Ph.D. and Dr. of Sci. theses in three scientific specialties and
corresponding scientific disciplines in which academic degrees are awarded:

05.13.11 — Mathematical Support and Software for Computers, Computing Complexes and
Computer Networks,

05.13.17 — Theoretical foundations of Informatics.

05.13.18 — Mathematical Modeling, Numerical Methods and Software Complexes.

All rights reserved. The journal contents may only be used by the permission of editors and authors.

© Institute of Computational Mathematics and Mathematical Geophysics of Siberian Branch of
Russian Academy of Sciences, 2021

© Institute of Information and Computing Technologies of the Ministry of Education of the
Republic of Kazakhstan, 2021

ПРЕДИСЛОВИЕ

В настоящем выпуске журнала „Проблемы информатики“ публикуются статьи, рекомендованные программным комитетом конференции „Математическое моделирование и суперкомпьютерные технологии“ и отражающие основные направления ее работы.

Международная конференция и молодежная школа „Математическое моделирование и суперкомпьютерные технологии“ проводилась 23–27 ноября 2020 г. в рамках Международного конгресса „Суперкомпьютерные дни в России“ на базе Нижегородского государственного университета им. Н. И. Лобачевского при поддержке корпорации Intel, компаний Huawei и группы компаний РСК. Основную тематику конференции составили:

- математическое моделирование динамики систем и процессов управления;
- теория динамических систем и бифуркаций;
- колебательные процессы в динамических сетях;
- модели и методы поддержки принятия решений;
- модели и методы искусственного интеллекта;
- математические модели в биологии и медицине;
- математическое моделирование природных процессов;
- технологии параллельных и распределенных вычислений;
- применение суперкомпьютерных технологий для решения вычислительно-сложных задач.

Основными задачами проведения мероприятия являлись представление актуальных результатов в области математического моделирования сложных процессов и явлений, обсуждение различных аспектов организации суперкомпьютерных вычислений, расширение контактов между специалистами для решения ресурсоемких прикладных задач, обмен опытом научно-образовательной деятельности при подготовке специалистов в области математического моделирования и параллельных вычислений.

Виктор Павлович Гергель, д-р. техн. наук, проф., директор Института информационных технологий, математики и кибернетики Нижегородского университета им. Н. И. Лобачевского, председатель орг. комитета XX Международной конференции и молодежной школы „Математическое моделирование и суперкомпьютерные технологии“

СОДЕРЖАНИЕ

Теоретическая и системная информатика

<i>Брикель Д. М., Ерофеев В. И.</i> Влияние поврежденности материала на параметры нелинейной изгибной и продольной волн, распространяющихся в балке	6
<i>Кулешов А. С., Соломина Д. В.</i> Применение алгоритма Ковачича для исследования задачи о качении тяжелого однородного шара по поверхности вращения	15

Прикладные информационные технологии

<i>Куприянова Т. В., Кислицын Д. И.</i> Применение методов машинного обучения в строительстве	25
<i>Петров С. С.</i> Конечно-элементная модель динамики морского льда и ее параллельная реализация с использованием библиотеки INMOST	36

Параллельное системное программирование и вычислительные технологии

<i>Исупов К. С., Князьев В. С., Бабешко И. П., Крутиков А. К.</i> Реализация и оценка производительности разреженного матрично-векторного умножения многократной точности на CUDA с использованием системы остаточных классов	49
<i>Чаплыгин А. В., Гусев А. В.</i> Гибридная модель мелкой воды с использованием технологий MPI-OpenMP	65

Правила представления и подготовки рукописей для публикации в журнале „ПРОБЛЕМЫ ИНФОРМАТИКИ“	83
---	----

CONTENTS

Theoretical informatics

<i>Brikkel D. M., Erofeev V. I.</i> Influence of material damage on the parameters of a nonlinear flexible and longitudinal wave which spread in a beam	6
<i>Kuleshov A. S., Solomina D. V.</i> Application of the Kovacic algorithm to the problem of rolling of a heavy homogeneous ball on a surface of revolution	15

Applied information technologies

<i>Kupriyanova T. V., Kislytsyn D. I.</i> Application of machine learning methods in building construction	25
<i>Petrov S. S.</i> Finite element model of sea ice dynamics and its parallel implementation using the INMOST library	36

Parallel system programming and computing technologies

<i>Isupov K., Knyazkov V., Babeshko I., Krutikov A.</i> Implementation and Performance Evaluation of Multiple Precision Sparse Matrix-Vector Multiplication on CUDA Using the Residue Number System	49
<i>Chaplygin A. V., Gusev A. V.</i> Shallow water model using a hybrid MPI/OpenMP parallel programming	65

Rules of presentation and preparation of manuscripts offered for publication	83
--	----

INFLUENCE OF MATERIAL DAMAGE ON THE PARAMETERS OF A NONLINEAR FLEXIBLE AND LONGITUDINAL WAVE WHICH SPREAD IN A BEAM

D. M. Brikkel*, V. I. Erofeev*,**

*National Research Lobachevsky State University of Nizhny Novgorod,
603950, Nizhny Novgorod, Russia

**Mechanical Engineering Research Institute of RAS,
603024, Nizhny Novgorod, Russia

DOI: 10.24411/2073-0667-2021-10001

As a rule, in the mechanics of solids, problems of dynamics are considered separately from problems of damage accumulation. When developing such methods, it is customary to postulate in advance that the elastic wave velocity is a given damage function, and then experimentally determine the proportionality coefficients. The phase wave velocity and wave attenuation are usually considered to be power functions of frequency and linear functions of damage. With its undoubted advantages (simplicity), this approach has a number of disadvantages, like any approach that is not based on mathematical models of processes and systems.

The authors of consider the problem to be self-consistent, including, in addition to the equation of damage development, the dynamic equation of the theory of elasticity. This approach made it possible to consider a number of applied problems of wave dynamics of damaged materials and structural elements.

Damage is usually understood as a reduction in the elastic response of the body due to a reduction in the effective area that transfers internal forces from one part of the body to another. Its part, caused, in turn, by the appearance and development of a scattered field of micro defects (micro cracks — in elasticity, dislocations — in plasticity, micro pores — during creep, surface micro cracks — during fatigue).

Not directly measurable (such as speed, force or temperature), damage, i. e. degradation of the mechanical properties of the body can be detected by analyzing the body's response to various external influences. According to experimental practice, the presence of a damage field in materials can be indirectly detected and partly quantitatively represented through a decrease in the ultrasonic signal propagation rate, a decrease in Young's modulus („modulus defect“), and a decrease in density („loosening“), change in hardness, drop in electric potential, drop in stress amplitude during cyclic testing, acceleration of creep in the third stage.

In traditional calculations, the scalar damageability parameter $\psi(x, t)$, which characterizes the relative density of micro defects uniformly dispersed per unit volume, is taken as a measure of damage during deformation development. This parameter is zero when there is no damage and is close to unity at the moment of failure.

An approach is proposed that allows one to obtain new dependences of the nonlinear flexural and longitudinal waves parameters on the material damage degree. In particular, the problem of the formation of intense bending waves of a stationary profile is considered within the framework of a geometrically nonlinear model of a damaged beam, as well as the problem of the formation of longitudinal waves, where geometric and physical nonlinearities are additionally taken into account.

During the study, when nonlinear flexural waves appear in an element, the presence of both periodic and solitary (localized in space) non-sinusoidal waves was determined. It is shown that the amplitudes of solitary and periodic waves increase with an increase in the damage parameter, and the length of the periodic and the width of a solitary wave decrease with an increase in the considered parameter. When nonlinear longitudinal waves appear in the element, it is proved that the width of the soliton decreases with decreasing damage parameter, in turn, the soliton velocity increases linearly with increasing damage parameter. New dependences of wave parameters (amplitude, width, wavelength) are determined taking into account their geometric nonlinearity on the parameters of material damage. This work is a continuation of a number of articles on a new approach that allows formulating a mathematical model describing the propagation of waves in elements, taking into account the accumulation of damage in their materials.

Key words: Bending wave, longitudinal wave, Material damage, material damage parameter, geometric nonlinearity, physical nonlinearity.

References

1. Kachanov L. M. Osnovy mekhaniki razrusheniya. M.: Nauka. 1974. 311 p.
2. Rabotnov Y. N. Polzuchest' elementov konstrukcij. M.: Nauka. 1966. 752 p.
3. Stulov A., Erofeev V. Frequency-dependent attenuation and phase velocity dispersion of an acoustic wave propagating in the media with damages // Generalized Continua as Models for Classical and Advanced Materials. Series: Advances Structured Materials, Altenbach, Holm, Forest, Samuel (Eds.), Springer, Switzerland. 2016. V. 42. P. 413–423.
4. Moiseev N. N. Asimptoticheskie metody nelinejnoj mekhaniki. M.: Nauka. 1981. 400 p.
5. Porubov A. V. Lokalizaciya nelinejnyh voln deformacii. M.: Fizmatlit. 2009. 208 p.



ВЛИЯНИЕ ПОВРЕЖДЕННОСТИ МАТЕРИАЛА НА ПАРАМЕТРЫ НЕЛИНЕЙНОЙ ИЗГИБНОЙ И ПРОДОЛЬНОЙ ВОЛН, РАСПРОСТРАНЯЮЩИХСЯ В БАЛКЕ

Д. М. Брикель*, В. И. Ерофеев*,**

*Нижегородский государственный университет им. Н.И. Лобачевского,
603950, Нижний Новгород, Россия

**Институт проблем машиностроения РАН,
603024, Нижний Новгород, Россия

УДК 539.3

DOI: 10.24411/2073-0667-2021-10001

Предлагается подход, позволяющий получить новые зависимости параметров нелинейных изгибных и продольных волн от степени поврежденности материала. В частности, рассмотрены задача о формировании интенсивных изгибных волн стационарного профиля в рамках геометрически нелинейной модели поврежденной балки, а также задача о формировании продольных волн, где дополнительно учитываются геометрическая и физическая нелинейности. В ходе исследования, при возникновении в элементе нелинейных изгибных волн, определено наличие как периодических, так и уединенных (локализованных в пространстве) несинусоидальных волн. Показано, что амплитуды уединенных и периодических волн увеличиваются с ростом параметра поврежденности, а длина периодической и ширина уединенной волн уменьшаются с ростом рассматриваемого параметра. При возникновении в элементе нелинейных продольных волн доказано, что ширина солитона уменьшается при уменьшении параметра поврежденности, в свою очередь, скорость солитона линейно увеличивается с ростом параметра поврежденности. Определены новые зависимости волновых параметров (амплитуда, ширина, длина волны) с учетом их геометрической нелинейности от параметров поврежденности материала. Данная работа является продолжением ряда статей о новом подходе, позволяющем сформулировать математическую модель, описывающую распространение волн в элементах с учетом накопления повреждений в их материалах.

Ключевые слова: изгибная волна, продольная волна, параметр поврежденности материала, геометрическая нелинейность, физическая нелинейность.

Введение. Рассматривается элемент, представленный в виде балки и совершающий поперечные (изгибные) и продольные колебания.

Считаем, что в материале балки могла накопиться поврежденность вследствие воздействия на нее статических и циклических нагрузок. Для описания меры поврежденности введем функцию Качанова-Работнова $\psi(x,t)$, значения которой равны нулю, когда повреждения отсутствуют, и близко к единице в момент разрушения материала [1, 2].

Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (Грант № 19-38-90282).

1. Изгибные колебания.

1.1. *Математическая модель изгибных колебаний.* Считая, что центральная ось балки остается нерастяжимой, дополнительно учитываем нелинейную связь деформации и перемещения, возникающую при изгибе (т. е. геометрическую нелинейность).

Обозначим перемещение частиц центральной оси элемента при изгибе через функцию $W(x,t)$. С учетом накопления повреждений в материале балки, ее динамика описывается системой уравнений:

$$\begin{cases} \frac{\partial^2 W}{\partial t^2} + c_s^2 r_y^2 \frac{\partial^4 W}{\partial x^4} - \frac{c_s^2}{2} \frac{\partial}{\partial x} \left[\left(\frac{\partial W}{\partial x} \right)^3 \right] = \beta_1 \frac{\partial \Psi}{\partial x} \\ \frac{\partial \Psi}{\partial t} + \alpha \Psi = \beta_2 E \frac{\partial W}{\partial x}. \end{cases} \quad (1)$$

Здесь $c_s = \sqrt{E\rho^{-1}}$, E — Модуль Юнга, ρ — плотность материала, J_y — осевой момент инерции, F — площадь поперечного сечения балки, α, β_1, β_2 — константы, характеризующие поврежденность материала (среди которых, где T — время релаксации [3], физический смысл других коэффициентов не столь очевиден; $\beta_1, \beta_2 < 0$). Система уравнений (1) сводится к одному нелинейному уравнению относительно поперечного перемещения W , которое в безразмерных переменных запишется в следующем виде:

$$\begin{aligned} \frac{\partial^2 U}{\partial \tau^2} + a_1 \frac{\partial^2 U}{\partial z^2} + a_2 \frac{\partial^3 U}{\partial \tau^3} + \frac{\partial^4 U}{\partial z^4} + a_2 \frac{\partial^5 U}{\partial z^4 \partial \tau} - \\ - \frac{a_3}{2} \frac{\partial}{\partial z} \left[\left(\frac{\partial U}{\partial z} \right)^3 \right] - \frac{a_2 a_3}{2} \frac{\partial^2}{\partial z \partial \tau} \left[\left(\frac{\partial U}{\partial z} \right)^3 \right] = 0 \end{aligned} \quad (2)$$

где $a_1 = -\beta_1 \beta_2 E (\alpha c_s^2)^{-1}$, $a_2 = c_s (r_y \alpha)^{-1}$, $a_3 = W_0^2 (r_y^2)^{-1}$.

Решение этого уравнения будем искать в виде бегущей волны стационарного профиля:

$$U = U(\xi), \quad (3)$$

где $\xi = z - V\tau$ — „бегущая“ координата, $V = const$ — скорость волны. При подстановке (3) в (2), исходя из условий, что $T \rightarrow 0$ (при малых значениях времени релаксации), т. е. $\alpha \rightarrow \infty$, решение последнего сводится к уравнению колебаний осциллятора Дуффинга [4]:

$$\frac{d^2 \Theta}{d\xi^2} + m_1 \Theta + m_2 \Theta^3 = 0 \quad (4)$$

Здесь — $\Theta = \frac{\partial W}{\partial \xi}$ — угол поворота поперечного сечения балки; $m_1 = a_1 + V^2$; $m_2 = \frac{-a_3}{2}$.

Знаки коэффициентов $m_1 m_2$ характеризуют возможность существования нелинейных стационарных изгибных волн. Первый коэффициент всегда положителен ($m_1 > 0$), а второй — всегда отрицателен ($m_2 < 0$).

По фазовой плоскости нелинейного осциллятора можно судить о существовании в данной системе как периодических (движения по сепаратрисе вокруг устойчивого положения равновесия), так и уединенных стационарных волн (движение по сепаратрисе, идущей из „седла“ в „седло“).

1.2. *Анализ зависимостей.* Периодическая волна описывается эллиптическим синусом (синусоидальное колебание) [4], форма которого близка к меандру

$$\Theta = Asn(k\xi, S), \quad (5)$$

где A — амплитуда волны; $k = (0,5(2m_1 + m_2A^2))^{\frac{1}{2}}$ — волновое число; $S^2 = m_2A^2(2m_1 + m_2A^2)^{-1}$ — квадрат модуля эллиптической функции, изменяющейся в интервале $0 \leq S^2 \leq 1$. Выразим амплитуду волны A и волновое число k через параметры исходной задачи:

$$A = \sqrt{\frac{4 \left(\frac{-\beta_1\beta_2E}{\alpha c_s^2} + V^2 \right) r_y^2 S^2}{S_0^2(1 + S^2)}}, \quad (6)$$

$$k = \sqrt{\frac{\left(\frac{-\beta_1\beta_2E}{\alpha c_s^2} + V^2 \right)}{(1 + S^2)}}, \quad (7)$$

Уединенная стационарная волна имеет форму перепада (кинка) и описывается гиперболическим тангенсом:

$$\Theta(\xi) = A^{(c)} th \left(\frac{\xi}{\Delta} \right), \quad (8)$$

Здесь

$$\Delta = \sqrt{\frac{2}{\left(\frac{-\beta_1\beta_2E}{\alpha c_s^2} + V^2 \right)}} \quad (9)$$

— ширина волны,

$$A^{(c)} = \pm \sqrt{\frac{2 \left(\frac{-\beta_1\beta_2E}{\alpha c_s^2} + V^2 \right) r_y^2}{W_0^2}} \quad (10)$$

— ее амплитуда.

Соотношения (6) и (10) показывают рост амплитуды периодической волны при увеличении параметра поврежденности материала. Длина периодической волны $\Lambda \sim k^{-1}$ (см. (7)) и ширина уединенной волны (см. (9)) уменьшаются с ростом параметра поврежденности.

Примечательно, что отношение амплитуды стационарной волны к волновому числу является величиной постоянной

$$\frac{A}{k} = \frac{2r_y S}{W_0} = const, \quad (11)$$

определенной только радиусом инерции поперечного сечения балки.

Заметим, что произведение амплитуды волны на ее ширину тоже является постоянной величиной

$$A^{(c)} \Delta = \frac{2r_y}{W_0} = const. \quad (12)$$

Выражения (11) и (12) тождественны при $S = 1$, поскольку при этом значении график эллиптического синуса переходит в гиперболический тангенс. Для балки кругового поперечного сечения осевой радиус инерции равен половине радиуса. Уравнение (12) в этом случае может быть выражено в виде $2A^{(c)} \Delta = \frac{d}{W_0}$, где d — диаметр балки.

2. Продольные колебания.

2.1 *Математическая модель продольных колебаний.* Динамика балки описывается моделью Бишопа. В заданной модели дополнительно учитываются нелинейности: геометрическая (нелинейная связь деформации и перемещения) и физическая (нелинейный закон Гука). Обозначим через $u(x, t)$ продольное перемещение ее срединной линии.

Нелинейная динамика стержня с учетом поврежденности его материала описывается системой уравнений:

$$\begin{cases} \frac{\partial^2 u}{\partial t^2} - c_0^2 \left(1 + \frac{\alpha_0}{E} \frac{\partial u}{\partial x} \right) \frac{\partial^2 u}{\partial x^2} - \nu^2 r_0^2 \frac{\partial^2}{\partial x^2} \left(\frac{\partial^2 u}{\partial t^2} - c_\tau^2 \frac{\partial^2 u}{\partial x^2} \right) = \beta_1 \frac{\partial \Psi}{\partial x} \\ \frac{\partial \Psi}{\partial t} + \alpha \Psi = \beta_2 E \frac{\partial u}{\partial x}. \end{cases} \quad (13)$$

Здесь $c_0 = \sqrt{E\rho^{-1}}$ — скорость, с которой распространялась бы продольная упругая волна в материале стержневого образца, если в нем не было бы повреждений; E — модуль Юнга; ρ — плотность материала, $r_0 = \sqrt{I_0 F_{-1}}$ — полярный радиус инерции поперечного сечения стержня, F — площадь поперечного сечения, $c_\tau = \sqrt{G\rho^{-1}}$ — скорость распространения упругой сдвиговой волны, G — модуль сдвига. Через $\alpha_0 = 3E + \nu_1(1 - 6\nu) + 6\nu_2(1 - 2\nu) + 2\nu_3$ обозначен коэффициент, характеризующий геометрическую и физическую упругие нелинейности стержня; $\nu_{1,2,3}$ — упругие модули Ламе третьего порядка.

Система (13) сводится к одному уравнению относительно продольного перемещения $u(x, t)$, которое имеет вид:

$$\begin{aligned} & \frac{\partial^2 u}{\partial t^2} - c_0^2 \left(1 + \frac{\beta_1 \beta_2 E}{\alpha c_0^2} \right) \frac{\partial^2 u}{\partial x^2} - \nu^2 r_0^2 \frac{\partial^2}{\partial x^2} \left(\frac{\partial^2 u}{\partial t^2} - c_\tau^2 \frac{\partial^2 u}{\partial x^2} \right) + \\ & + \frac{1}{\alpha} \frac{\partial^3 u}{\partial t^3} - \frac{c_0^2}{\alpha} \frac{\partial^3 u}{\partial x^2 \partial t} - \frac{\nu^2 r_0^2}{\alpha} \left(\frac{\partial^5 u}{\partial x^2 \partial t^3} - c_\tau^2 \frac{\partial^5 u}{\partial x^4 \partial t} \right) - \\ & - \frac{6c_0^2 \alpha_0}{E} \frac{\partial u}{\partial x} \frac{\partial^2 u}{\partial x^2} - \frac{6c_0^2 \alpha_0}{E \alpha} \frac{\partial}{\partial t} \left(\frac{\partial u}{\partial x} \frac{\partial^2 u}{\partial x^2} \right) = 0 \end{aligned} \quad (14)$$

Предположим, что затухание волны с учетом поврежденности материала и нелинейность являются величинами одного порядка малости $\varepsilon = A|\alpha_0|(E\Lambda)^{-1}$, (A — амплитуда волны, Λ — длина волны).

Решая уравнение (14) в виде асимптотического разложения по малому параметру,

$$u = u_0 + \varepsilon u_1 + \dots \quad (15)$$

введем дополнительно новые переменные.

$$z = x - ct; \tau = \varepsilon t \quad (16)$$

После подстановки (15) и (16) в (14) в нулевом приближении по ε , получим выражение для скорости

$$c = c_0 \sqrt{1 + \frac{\beta_1 \beta_2 E}{\alpha c_0^2}} \quad (17)$$

В первом приближении по ε получим эволюционное уравнение относительно осевой деформации $U = \frac{\partial u_0}{\partial z}$:

$$\frac{\partial U}{\partial \tau} + a_1 \frac{\partial^3 U}{\partial z^3} + a_2 \frac{\partial^2 U}{\partial z^2} - a_3 \frac{\partial^4 U}{\partial z^4} + \\ + \alpha_1 U \frac{\partial U}{\partial z} - \alpha_2 \frac{\partial}{\partial z} \left(U \frac{\partial U}{\partial z} \right) = 0 \quad (18)$$

Здесь введены обозначения:

$$a_1 = \frac{\nu^2 r_0^2 c_0 \left(1 + \frac{\beta_1 \beta_2 E}{\alpha c_0^2} - \frac{c_0^2}{c_0^2} \right)}{2\varepsilon \sqrt{1 + \frac{\beta_1 \beta_2 E}{\alpha c_0^2}}}, a_2 = \frac{\beta_1 \beta_2 E}{2\varepsilon \alpha^2}, a_3 = \frac{\nu^2 r_0^2 c_0 \left(1 + \frac{\beta_1 \beta_2 E}{\alpha c_0^2} - \frac{c_0^2}{c_0^2} \right)}{2\varepsilon \alpha}, \quad (19)$$

$$\alpha_1 = \frac{3c_0 \alpha_0}{\varepsilon E \sqrt{1 + \frac{\beta_1 \beta_2 E}{\alpha c_0^2}}}, \alpha_2 = \frac{3c_0^2 \alpha_0}{\varepsilon E \alpha}. \quad (20)$$

Заметим, что полученное уравнение (18) обобщает уравнение Кортевега-де Вриза-Бюргерса [5] одним нелинейным (с коэффициентом α_2) и одним диссипативным (с коэффициентом α_3) слагаемыми.

2.2. Анализ зависимостей. Уравнение (18) имеет точное аналитическое решение, выражающееся через гиперболический косинус:

$$U = A ch^{-2} \left(\frac{z - V\tau}{\Delta} \right) + B, \quad (21)$$

где

$$A = \frac{12a_3}{\alpha_2 \Delta^2}, B = \frac{\left(a_2 - \frac{4a_3}{\Delta^2} \right)}{\alpha_2}, V = \frac{\alpha_1 a_2}{\alpha_2}, a_1 = \frac{\alpha_1 a_3}{\alpha_2}. \quad (22)$$

Соотношения между параметрами (22) находятся непосредственной подстановкой выражения (21) в уравнение (18). Из (19), (20) следует, что последнее соотношение в (22) выполняется тождественно.

График функции (21) имеет колоколообразную форму и характеризует нелинейную уединенную стационарную волну — солитон деформации, распространяющийся со скоростью V , имеющий амплитуду и ширину Δ . Физически реализуемым является лишь тот случай, когда в волне деформации отсутствует постоянная составляющая. Рассмотрение такого случая заставляет добавить к соотношениям (22) еще одно условие $B = 0$, что приводит к следующей зависимости ширины солитона от параметров (19):

$$\Delta^2 = \frac{4a_3}{a_2}. \quad (23)$$

Соотношения (19), (20), (22) и (23) позволяют установить, как зависят параметры солитона от коэффициента $\gamma = \beta_1 \beta_2 E(\alpha)^{-1}$, характеризующего поврежденность материала.

Амплитуда солитона

$$A = \frac{\gamma \rho}{2\alpha_0} \quad (24)$$

линейно растет с увеличением γ .

Ширина солитона определяется соотношением

$$\Delta = \frac{2\nu r_0 c_0 \sqrt{1 + \frac{\gamma}{c_0^2} - \frac{c_t^2}{c_0^2}}}{\sqrt{\gamma}}, \quad (25)$$

а его скорость — соотношением

$$V = \frac{\gamma}{2\varepsilon c_0 \sqrt{1 + \frac{\gamma}{c_0^2}}}. \quad (26)$$

Поскольку $\frac{\gamma}{c_0^2}$, то из (25) следует, что ширина солитона (Δ) будет уменьшаться как $\frac{1}{\gamma}$, а из (26) следует, что скорость солитона (V) будет линейно увеличиваться с ростом γ .

Заключение. Развиваемый подход позволяет сформулировать и решить самосогласованную задачу, характеризуемую совместным решением уравнения динамики балки и кинетического уравнения поврежденности ее материала. При экспериментальном определении характеристик поврежденности материала данная работа может найти применение при разработке методик акустического контроля длительно эксплуатируемых элементов конструкций.

Список литературы

1. Качанов Л. М. Основы механики разрушения. М.: Наука. 1974. 311с.
2. Работнов Ю. Н. Ползучесть элементов конструкций. М.: Наука. 1966. 752 с.
3. Stulov A., Erofeev V. Frequency-dependent attenuation and phase velocity dispersion of an acoustic wave propagating in the media with damages // Generalized Continua as Models for Classical and Advanced Materials. Series: Advances Structured Materials, Altenbach, Holm, Forest, Samuel (Eds.), Springer, Switzerland. 2016. V. 42. P. 413–423.
4. Моисеев Н. Н. Асимптотические методы нелинейной механики. М.: Наука. 1981. 400 с.
5. Порубов А. В. Локализация нелинейных волн деформации. М.: Физматлит. 2009. 208 с.



Бриккель Дмитрий Максимович — аспирант и мл. науч. сотр. Института информационных технологий математики и механики Нижегородского государственного университета им. Н.И. Лобачевского, e-mail: archive.94@mail.ru, тел.: +79082351064.

Магистр по направлению „Строительство“ Нижегородского государственного архитектурно-строительного университета. Аспирант по направлению „Механика деформируемого твердого тела“ Нижегородского государственного университета им. Н.И. Лобачевского. Специализируется в области волновой динамики элементов конструкций с учетом накопления повреждений в их материалах.

Brikkel D. M. postgraduate student and research assistant of the institute of Information Technologies of Mathematics and

Mechanicsof in Lobachevsky State University of Nizhny Novgorod, e-mail: archive.94@mail.ru, Tel.:+79082351064.

Master's degree in „Construction“ of the Nizhny Novgorod State University of Architecture and Civil Engineering. Postgraduate student in „Mechanics of Deformable Solid“ of the Lobachevsky State University of Nizhny Novgorod. He specializes in „Wave dynamics of structural elements, taking into account the accumulation of damage in their materials“.



Ерофеев Владимир Иванович — д-р физ.-мат. наук, директор Института проблем машиностроения РАН, профессор Нижегородского государственного университета им. Н.И. Лобачевского, e-mail: erof.vi@yandex.ru, тел.:+79103843528.

Специалист в области волновой динамики механических систем, в числе достижений: исследована устойчивость движения высокоскоростных объектов по рельсовым направляющим ракетного трека; разработан метод, создана и внедрена спектрально-акустическая система диагностики материалов и конструкций, развиты научные основы систем виброзащиты машин с использованием инерционности магнитореологических сред.

Erofeev V. I. doctor of physical and mathematical sciences, director of the Mechanical Engineering Research Institute of the RAS,

professor of the Lobachevsky State University of Nizhny Novgorod, e-mail: erof.vi@yandex.ru, tel.: +79103843528.

A specialist in the field of wave dynamics of mechanical systems, among the achievements: the stability of the movement of high-speed objects along the rails of a rocket track was investigated; a method has been developed, a spectral-acoustic system for diagnostics of materials and structures has been created and implemented, the scientific foundations of vibration protection systems for machines using the inertia of magnetorheological media have been developed.

Дата поступления — 03.02.2021

APPLICATION OF THE KOVACIC ALGORITHM TO THE PROBLEM OF ROLLING OF A HEAVY HOMOGENEOUS BALL ON A SURFACE OF REVOLUTION

A. S. Kuleshov, D. V. Solomina

Department of Mechanics and Mathematics, Lomonosov Moscow State University
119234, Moscow, Russia

DOI: 10.24411/2073-0667-2021-10002

Problems of description of the motion of bodies that roll on a fixed or moving rigid surface have a long history. They are closely related to the process of formation and development of a large branch of analytical mechanics, namely, dynamics of nonholonomic systems. In works of I. Newton, L. Euler, I. Bernoulli, J. D'Alembert and J. Lagrange, some problems on the rolling of rigid bodies without sliding were studied; these problems are typical in the dynamics of systems with nonholonomic constraints and hence they are considered as classical problems of nonholonomic mechanics.

One of such classical problems is the problem of rolling without sliding of a homogeneous ball on a fixed surface under the action of gravity. For the first time this problem has been studied by E. J. Routh in his classical treatise „The Advanced Part of a Treatise on the Dynamics of a System of Rigid Bodies“. Routh derived the equations of motion of the ball in semifixed axes and indicated the cases of their integrability. In particular, Routh firstly proved that a homogeneous ball, moving within a vertical cylinder under the action of gravity, does not tending downwards on the average. This physical fact may be observed while playing basketball, when the ball has almost hit the basket, but then rapidly jumps out of it, suddenly lifting upwards. Actually, the majority of the subsequent authors were just stating the results, obtained by Routh, without any essential additions. In particular, this problem has been considered in details by Ju. I. Neimark and N. A. Fufaev in his classical monograph „Dynamics of Nonholonomic Systems“.

Usually, when considering this problem, following the E. J. Routh approach it is convenient to define explicitly the surface, on which the ball's centre of gravity is moving and not the supporting surface along which the ball rolls. The surface, on which the ball's centre of gravity is moving, is equidistant to the surface, over which the ball rolls. From the classical works of E. J. Routh and F. Noether it was known that if the ball rolls on a surface such that its centre of gravity moves along a surface of revolution, then the problem is reduced to solving the second order linear differential equation with respect to the projection of velocity of the ball's centre of gravity onto the tangent to the parallel of the corresponding surface of revolution. However it is impossible to find the general solution of the corresponding second order linear differential equation for the arbitrary surface of revolution in explicit form. Therefore it is interesting to study for which surface of revolution the general solution of the corresponding second order linear differential equation can be expressed explicitly, in particular, in terms of liouvillian functions. Recall that liouvillian functions are functions constructed from rational functions by algebraic operations, taking exponentials, and integration. To solve this problem it is possible to apply the so-called Kovacic algorithm to the corresponding second order linear differential equation. In 1986, the American mathematician J. Kovacic presented a very effective algorithm for finding a general solution of a second order linear differential equation with variable coefficients for the case where this solution can be expressed in terms of liouvillian functions. If a linear differential

equation has no liouvillian solutions, the Kovacic algorithm also allows one to ascertain this fact. The necessary condition for application of the Kovacic algorithm to a second order linear differential equation is that the coefficients of this equation should be rational functions of independent variable. There are no new ideas in the algorithm. It is simply brute-force calculations. The algorithm does require that the partial fraction expansion of the coefficients of the differential equation be known, thus one needs to factor a polynomial in one independent variable over the complex numbers into linear factors. Once the partial fraction expansions are known, only linear algebra is required.

In this paper we apply the Kovacic algorithm to the problem of rolling of a heavy homogeneous ball on a fixed surface such that the centre of gravity of the ball moves along a given surface of revolution. We present our own method to derive the corresponding second order linear differential equation, the integration of which leads to the solution of the problem. In the case when the centre of gravity of the ball moves along the paraboloid of revolution we are presenting the corresponding linear differential equation in explicit form and reduce its coefficients to a form of rational functions. Using the Kovacic algorithm we prove that the general solution of the corresponding second order linear differential equation is expressed in terms of liouvillian functions for all values of parameters of the problem.

Key words: rolling without sliding; homogeneous ball; surface of revolution; Kovacic algorithm; Liouvillian solutions.

References

1. Routh E. J. *The Advanced Part of a Treatise on the Dynamics of a System of Rigid Bodies: Being Part II of a Treatise on the Whole Subject*. Cambridge: Cambridge University Press, 2013.
2. Noether F. *Über rollende Bewegung einer Kugel auf Rotationsflächen*. Leipzig: Teubner, 1909.
3. Kovacic J. An algorithm for solving second order linear homogeneous differential equations // *J. Symb. Comput.* 1986. Vol. 2. Issue 1. P. 3–43.
4. Kaplansky I. *An Introduction to Differential Algebra*. Paris: Hermann, 1957.
5. Do Carmo M. P. *Differential geometry of curves and surfaces*. New Jersey: Prentice-Hall, Englewood Cliffs, 1976.



ПРИМЕНЕНИЕ АЛГОРИТМА КОВАЧИЧА ДЛЯ ИССЛЕДОВАНИЯ ЗАДАЧИ О КАЧЕНИИ ТЯЖЕЛОГО ОДНОРОДНОГО ШАРА ПО ПОВЕРХНОСТИ ВРАЩЕНИЯ

А. С. Кулешов, Д. В. Соломина

Механико-математический факультет МГУ им. М. В. Ломоносова,
119234, Москва, Россия

УДК 531.384+517.926.4

DOI: 10.24411/2073-0667-2021-10002

Задача о качении без скольжения однородного шара по неподвижной поверхности под действием силы тяжести является одной из классических задач механики неголономных систем. Обычно при рассмотрении этой задачи, следуя подходу, предложенному в трактате Э. Дж. Рауса [1], принято задавать в явном виде поверхность, по которой движется центр шара, а не опорную поверхность, по которой катится шар. Поверхность, по которой движется центр шара, является эквидистантной к поверхности, по которой движется точка контакта. Еще из работ Э. Дж. Рауса [1] и Ф. Нетера [2] было известно, что если при качении шара по поверхности под действием силы тяжести его центр движется по поверхности вращения, то задача сводится к интегрированию одного линейного дифференциального уравнения второго порядка относительно компоненты скорости центра шара в проекции на направление касательной к параллели поверхности вращения. В общем случае (для произвольной поверхности вращения) получить решение этого уравнения в явном виде невозможно. Поэтому представляет интерес вопрос, для каких поверхностей вращения соответствующее линейное дифференциальное уравнение второго порядка допускает общее решение, выражющееся в явном виде, например, с помощью лиувиллевых функций. Лиувиллевы функции — это функции, которые строятся последовательно из рациональных функций с использованием алгебраических операций, неопределенного интегрирования и взятия экспоненты заданного выражения [3, 4]. Необходимые и достаточные условия существования решения линейного дифференциального уравнения второго порядка, выражющегося через лиувиллевы функции, дает так называемый алгоритм Ковачича [4]. В данной работе мы приводим наш собственный способ получения линейного дифференциального уравнения второго порядка, к интегрированию которого сводится задача о качении тяжелого шара по неподвижной поверхности, такой что центр шара при качении движется по заданной поверхности вращения. Затем при помощи замены независимой переменной мы приводим коэффициенты этого уравнения к виду рациональных функций. Применяя затем к полученному линейному дифференциальному уравнению второго порядка алгоритм Ковачича, мы показываем, что для случая, когда центр шара принадлежит параболоиду вращения, общее решение данного уравнения выражается через лиувиллевы функции.

Ключевые слова: качение без проскальзывания; однородный шар; поверхность вращения; алгоритм Ковачича; лиувиллевы решения.

Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (грант № 19-01-00140 и грант № 20-01-00637).

Постановка задачи. Рассмотрим задачу о движении однородного шара по произвольной абсолютно шероховатой поверхности под действием сил, результирующая которых проходит через центр шара [1]. Пусть G — центр тяжести шара, а подвижными осями GC , GA , GB будут соответственно нормаль к опорной поверхности и две перпендикулярные прямые, лежащие в касательной плоскости, построенной в точке соприкосновения шара с поверхностью. Направления прямых GA и GB определим позднее. Обозначим через \mathbf{e}_1 , \mathbf{e}_2 , \mathbf{e}_3 единичные базисные векторы осей GA , GB и GC соответственно. Пусть $\boldsymbol{\Omega} = \theta_1\mathbf{e}_1 + \theta_2\mathbf{e}_2 + \theta_3\mathbf{e}_3$ — угловая скорость выбранной подвижной системы координат; $\mathbf{v}_G = u\mathbf{e}_1 + v\mathbf{e}_2 + w\mathbf{e}_3$ — вектор скорости точки G (причем очевидно, что $w = 0$, поскольку шар не отрывается от опорной поверхности во все время движения); $\omega = \omega_1\mathbf{e}_1 + \omega_2\mathbf{e}_2 + \omega_3\mathbf{e}_3$ — угловая скорость вращения шара в проекции на оси рассматриваемой системы координат. Обозначим через $\mathbf{R} = F\mathbf{e}_1 + F'\mathbf{e}_2 + R\mathbf{e}_3$ силу реакции, действующей на шар со стороны опорной поверхности. Через $\mathbf{P} = X\mathbf{e}_1 + Y\mathbf{e}_2 + P\mathbf{e}_3$ обозначим результирующую силу, приложенную к центру масс шара. Пусть m — масса шара, a — его радиус, J — момент инерции шара относительно любой оси, проходящей через его центр тяжести (совпадающей с геометрическим центром шара). Предполагая, что шар катится по выпуклой стороне неподвижной поверхности, и что положительное направление оси GC направлено наружу в сторону выпуклости, запишем уравнения движения шара в векторном виде:

$$m\dot{\mathbf{v}}_G + [\boldsymbol{\Omega} \times \mathbf{v}_G] = \mathbf{P} + \mathbf{R}, \quad (1)$$

$$J\dot{\omega} + [\boldsymbol{\Omega} \times J\omega] = [\overrightarrow{GK} \times \mathbf{R}]. \quad (2)$$

Уравнения (1) и (2) выражают, соответственно, законы изменения импульса и кинетического момента шара относительно выбранной подвижной системы координат. Здесь $\overrightarrow{GK} = -a\mathbf{e}_3$ — радиус-вектор из центра масс G шара в точку касания с опорной поверхностью. Поскольку скорость точки шара, находящейся в соприкосновении с опорной поверхностью, в каждый момент времени равна нулю, то имеем:

$$\mathbf{v}_G + [\omega \times \overrightarrow{GK}] = 0. \quad (3)$$

В скалярной форме уравнения (1)–(3) записываются следующим образом:

$$m\ddot{u} - m\theta_3v = X + F, \quad m\dot{v} + m\theta_3u = Y + F', \quad m\theta_1v - m\theta_2u = P + R; \quad (4)$$

$$J\dot{\omega}_1 + J\theta_2\omega_3 - J\theta_3\omega_2 = F'a, \quad J\dot{\omega}_2 + J\theta_3\omega_1 - J\theta_1\omega_3 = -Fa, \quad \dot{\omega}_3 + \theta_1\omega_2 - \theta_2\omega_1 = 0; \quad (5)$$

$$u - a\omega_2 = 0, \quad v + a\omega_1 = 0. \quad (6)$$

Исключая F , F' , ω_1 , ω_2 из уравнений (4)–(6), получаем:

$$\dot{u} - \theta_3v = \frac{a^2X}{J + ma^2} + \frac{Ja\theta_1\omega_3}{J + ma^2}, \quad \dot{v} + \theta_3u = \frac{a^2Y}{J + ma^2} + \frac{Ja\theta_2\omega_3}{J + ma^2}. \quad (7)$$

Уравнения (7) можно рассматривать как уравнения такого движения центра масс шара, которое получается в предположении, что центр масс G движется по гладкой поверхности, в точке G приложены дополнительные силы

$$\frac{Ja\theta_1\omega_3}{J+ma^2} \quad \text{и} \quad \frac{Ja\theta_2\omega_3}{J+ma^2}$$

по осям GA и GB , а ранее приложенные силы уменьшены в отношении

$$\frac{a^2}{J+ma^2}.$$

Центр тяжести шара G движется по поверхности, полученной из данной поверхности смещением по нормали на расстояние, равное радиусу шара. Пусть оси GA и GB направлены по касательным к линиям кривизны этой поверхности. Найдем выражение для угловой скорости Ω выбранной подвижной системы координат GA, GB, GC в зависимости от компонент u и v скорости центра масс G шара. Будем считать, что поверхность, по которой движется центр шара, задается относительно некоторой неподвижной системы координат уравнением

$$\mathbf{r} = \mathbf{r}(q_1, q_2), \quad (8)$$

где q_1 и q_2 — гауссовые координаты на поверхности. Предположим, что координатная сеть на поверхности (8) составлена из линий кривизны. Направления этих линий в каждой точке указываются ортогональными единичными векторами

$$\mathbf{e}_1 = \frac{1}{h_1} \frac{\partial \mathbf{r}}{\partial q_1}, \quad \mathbf{e}_2 = \frac{1}{h_2} \frac{\partial \mathbf{r}}{\partial q_2}, \quad (\mathbf{e}_i \cdot \mathbf{e}_j) = \delta_{ij}. \quad (9)$$

Здесь через h_1, h_2 обозначены параметры Ламе

$$h_i(q_1, q_2) = \left| \frac{\partial \mathbf{r}}{\partial q_i} \right|, \quad i = 1, 2.$$

Вектор $\mathbf{e}_3 = [\mathbf{e}_1 \times \mathbf{e}_2]$ является вектором нормали к поверхности (8) в точке (q_1, q_2) . С другой стороны, скорость центра масс \mathbf{v}_G шара может быть определена по формуле:

$$\mathbf{v}_G = \frac{d\mathbf{r}}{dt} = \frac{\partial \mathbf{r}}{\partial q_1} \dot{q}_1 + \frac{\partial \mathbf{r}}{\partial q_2} \dot{q}_2 = u\mathbf{e}_1 + v\mathbf{e}_2.$$

Отсюда следует, что компоненты скорости u и v связаны с координатами q_1, q_2 и их производными формулами:

$$u = h_1 \dot{q}_1, \quad v = h_2 \dot{q}_2. \quad (10)$$

Обозначая через $k_i(q_1, q_2)$, $i = 1, 2$ главные кривизны поверхности (8), имеем:

$$\frac{\partial \mathbf{e}_3}{\partial q_1} = -h_1 k_1 \mathbf{e}_1, \quad \frac{\partial \mathbf{e}_3}{\partial q_2} = -h_2 k_2 \mathbf{e}_2. \quad (11)$$

Формулы (11) являются следствием известной в дифференциальной геометрии теоремы (формулы) Родрига [5], в которой дополнительно нужно учесть, что выбранная нами координатная сеть на поверхности (8) является ортогональной и составленной из линий кривизны. С помощью формул (9) и (11) можно получить следующие соотношения:

$$\begin{aligned} \frac{\partial \mathbf{e}_1}{\partial q_1} &= -\frac{1}{h_2} \frac{\partial h_1}{\partial q_2} \mathbf{e}_2 + h_1 k_1 \mathbf{e}_3, & \frac{\partial \mathbf{e}_1}{\partial q_2} &= \frac{1}{h_1} \frac{\partial h_2}{\partial q_1} \mathbf{e}_2, \\ \frac{\partial \mathbf{e}_2}{\partial q_1} &= \frac{1}{h_2} \frac{\partial h_1}{\partial q_2} \mathbf{e}_1, & \frac{\partial \mathbf{e}_2}{\partial q_2} &= -\frac{1}{h_1} \frac{\partial h_2}{\partial q_1} \mathbf{e}_1 + h_2 k_2 \mathbf{e}_3. \end{aligned} \quad (12)$$

Угловая скорость системы координат GA, GB, GC находится по стандартной формуле:

$$\boldsymbol{\Omega} = (\dot{\mathbf{e}}_2 \cdot \mathbf{e}_3) \mathbf{e}_1 + (\dot{\mathbf{e}}_3 \cdot \mathbf{e}_1) \mathbf{e}_2 + (\dot{\mathbf{e}}_1 \cdot \mathbf{e}_2) \mathbf{e}_3,$$

где обозначено

$$\dot{\mathbf{e}}_i = \frac{d\mathbf{e}_i}{dt} = \frac{\partial \mathbf{e}_i}{\partial q_1} \dot{q}_1 + \frac{\partial \mathbf{e}_i}{\partial q_2} \dot{q}_2, \quad i = 1, 2, 3.$$

Принимая во внимание формулы (11)–(12), для угловой скорости $\boldsymbol{\Omega}$ получим следующее выражение:

$$\boldsymbol{\Omega} = h_2 k_2 \dot{q}_2 \mathbf{e}_1 - h_1 k_1 \dot{q}_1 \mathbf{e}_2 + \left(\frac{\dot{q}_2}{h_1} \frac{\partial h_2}{\partial q_1} - \frac{\dot{q}_1}{h_2} \frac{\partial h_1}{\partial q_2} \right) \mathbf{e}_3.$$

Учитывая формулы (10), перепишем выражение для угловой скорости $\boldsymbol{\Omega}$ в виде:

$$\boldsymbol{\Omega} = k_2 v \mathbf{e}_1 - k_1 u \mathbf{e}_2 + \frac{1}{h_1 h_2} \left(\frac{\partial h_2}{\partial q_1} v - \frac{\partial h_1}{\partial q_2} u \right) \mathbf{e}_3.$$

Таким образом, для компонент $\theta_1, \theta_2, \theta_3$ угловой скорости $\boldsymbol{\Omega}$ подвижной системы координат GA, GB, GC мы имеем следующие выражения:

$$\theta_1 = k_2 v, \quad \theta_2 = -k_1 u, \quad \theta_3 = \frac{1}{h_1 h_2} \left(\frac{\partial h_2}{\partial q_1} v - \frac{\partial h_1}{\partial q_2} u \right). \quad (13)$$

Теперь предположим, что поверхность, по которой движется центр масс G шара, является поверхностью вращения, заданной относительно некоторой неподвижной системы координат уравнением

$$\mathbf{r} = \begin{pmatrix} \rho(q_1) \cos q_2 \\ \rho(q_1) \sin q_2 \\ \zeta(q_1) \end{pmatrix}. \quad (14)$$

В этом случае параметры Ламе h_1 и h_2 имеют вид:

$$h_1 = h_1(q_1) = \sqrt{\left(\frac{d\rho}{dq_1} \right)^2 + \left(\frac{d\zeta}{dq_1} \right)^2}, \quad h_2 = h_2(q_1) = \rho(q_1), \quad (15)$$

а главные кривизны k_1 и k_2 поверхности вычисляются по формулам:

$$k_1 = k_1(q_1) = \frac{\left(\frac{d^2\zeta}{dq_1^2} \frac{d\rho}{dq_1} - \frac{d\zeta}{dq_1} \frac{d^2\rho}{dq_1^2} \right)}{\left(\left(\frac{d\rho}{dq_1} \right)^2 + \left(\frac{d\zeta}{dq_1} \right)^2 \right)^{\frac{3}{2}}}, \quad k_2 = k_2(q_1) = \frac{\frac{d\zeta}{dq_1}}{\rho \sqrt{\left(\frac{d\rho}{dq_1} \right)^2 + \left(\frac{d\zeta}{dq_1} \right)^2}}. \quad (16)$$

Линиями кривизны на поверхности вращения являются ее меридианы и параллели. Пусть вертикальная ось Z является осью симметрии рассматриваемой поверхности вращения. Кроме координат q_1 и q_2 , введем углы Эйлера θ, ψ и φ так, что угол, который ось GC составляет с осью Z , равен $\pi/2 - \theta$, а ψ — угол, который плоскость, содержащая оси Z и GC , составляет с некоторой фиксированной вертикальной плоскостью. Будем считать, что компоненты $\theta_1, \theta_2, \theta_3$ угловой скорости $\boldsymbol{\Omega}$ системы координат GA, GB, GC , определяются при помощи стандартных кинематических формул Эйлера

$$\theta_1 = \dot{\psi} \sin \theta \sin \varphi + \dot{\theta} \cos \varphi, \quad \theta_2 = \dot{\psi} \sin \theta \cos \varphi - \dot{\theta} \sin \varphi, \quad \theta_3 = \dot{\psi} \cos \theta + \dot{\varphi},$$

в которых значение угла φ положено равным $-\pi/2$. Поэтому получаем:

$$\theta_1 = -\dot{\psi} \sin \theta, \quad \theta_2 = \dot{\theta}, \quad \theta_3 = \dot{\psi} \cos \theta. \quad (17)$$

С другой стороны, сравнивая полученные формулы с формулами (13), находим

$$-\dot{\psi} \sin \theta = k_2 h_2 \dot{q}_2, \quad \dot{\theta} = -k_1 h_1 \dot{q}_1, \quad \dot{\psi} \cos \theta = \frac{1}{h_1} \frac{dh_2}{dq_1} \dot{q}_2. \quad (18)$$

Из второго уравнения системы (18) определяется связь между переменными θ и q_1 . Поэтому можно считать, что поверхность (14) задана в зависимости от θ и q_2 , то есть

$$\rho|_{q_1=q_1(\theta)} = \sigma(\theta), \quad \zeta|_{q_1=q_1(\theta)} = \tau(\theta). \quad (19)$$

Параметры Ламе, вычисляемые согласно (15), определяются теперь формулами:

$$h_1 = h_1(\theta) = \sqrt{\left(\frac{d\sigma}{d\theta}\right)^2 + \left(\frac{d\tau}{d\theta}\right)^2}, \quad h_2 = h_2(\theta) = \sigma(\theta), \quad (20)$$

а главные кривизны $k_1 = k_1(\theta)$ и $k_2 = k_2(\theta)$ поверхности равны

$$k_1 = k_1(\theta) = \frac{\left(\frac{d^2\tau}{d\theta^2} \frac{d\sigma}{d\theta} - \frac{d\tau}{d\theta} \frac{d^2\sigma}{d\theta^2}\right)}{\left(\left(\frac{d\sigma}{d\theta}\right)^2 + \left(\frac{d\tau}{d\theta}\right)^2\right)^{\frac{3}{2}}}, \quad k_2 = k_2(\theta) = \frac{\frac{d\tau}{d\theta}}{\sigma \sqrt{\left(\frac{d\sigma}{d\theta}\right)^2 + \left(\frac{d\tau}{d\theta}\right)^2}}. \quad (21)$$

Теперь учтем вторую из формул (18), которую представим в виде $\dot{\theta} = -k_1 u$, и найдем из нее, что

$$u = -\frac{\dot{\theta}}{k_1}. \quad (22)$$

Теперь рассмотрим третью из формул (5) и формулы (6). Из формул (6) следует, что

$$\omega_1 = -\frac{v}{a}, \quad \omega_2 = \frac{u}{a} = -\frac{\dot{\theta}}{ak_1}.$$

Учитывая эти формулы, а также формулы (13) и (18), из третьего уравнения системы (5) получаем:

$$\dot{\omega}_3 = \theta_2 \omega_1 - \theta_1 \omega_2 = \frac{v \dot{\theta}}{ak_1} (k_2 - k_1). \quad (23)$$

Окончательно из формулы (23) получаем уравнение

$$\frac{d\omega_3}{d\theta} = \frac{v}{ak_1} (k_2 - k_1). \quad (24)$$

Теперь предположим, что качение шара происходит под действием силы тяжести. Тогда имеем:

$$Y = 0, \quad \theta_3 = \frac{1}{h_1 h_2} \frac{dh_2}{d\theta} v$$

и второе из уравнений (7) принимает вид:

$$\frac{dv}{d\theta} - \frac{v}{h_1 h_2 k_1} \frac{dh_2}{d\theta} = \frac{Ja}{J + ma^2} \omega_3. \quad (25)$$

Дифференцируя повторно формулу (25) и принимая во внимание уравнение (24), найдем:

$$\frac{d}{d\theta} \left(\frac{dv}{d\theta} - \frac{v}{h_1 h_2 k_1} \frac{dh_2}{d\theta} \right) = \frac{J}{J + ma^2} \frac{v}{k_1} (k_2 - k_1). \quad (26)$$

Таким образом, задача о качении шара по неподвижной выпуклой поверхности под действием силы тяжести в предположении, что центр тяжести G шара движется по поверхности вращения, сводится к интегрированию линейного дифференциального уравнения второго порядка (26) относительно компоненты скорости v центра тяжести шара. Коэффициенты данного уравнения определяются формой поверхности вращения, по которой движется центр тяжести шара. Можно поставить вопрос о том, для каких поверхностей вращения уравнение (26) интегрируется в явном виде, например, его общее решение выражается с помощью лиувиллевых функций. Лиувиллевы функции строятся последовательно из рациональных функций с использованием алгебраических операций, неопределенного интегрирования и взятия экспоненты заданного выражения [3, 4]. Для ответа на вопрос о существовании лиувиллевых решений у линейного дифференциального уравнения второго порядка обычно используется алгоритм Ковачича [3]. Ниже доказано, что задача о качении тяжелого шара по параболоиду вращения интегрируется в лиувиллевых функциях.

Качение по параболоиду вращения. Пусть абсолютно шероховатая поверхность, по которой катится шар, такова, что геометрическое место центров шара есть параболоид вращения с вертикальной осью, расположенный вершиной вверх. Уравнение параболоида запишем в виде (14):

$$\mathbf{r} = \begin{pmatrix} Rq_1 \cos q_2 \\ Rq_1 \sin q_2 \\ -\frac{Rq_1^2}{2} \end{pmatrix}.$$

В рассматриваемом случае

$$\rho(q_1) = Rq_1, \quad \zeta(q_1) = -\frac{Rq_1^2}{2}.$$

Здесь R — некоторый параметр, имеющий размерность длины. Параметры Ламе h_1 и h_2 , вычисляемые по формуле (15), имеют вид

$$h_1 = R\sqrt{1 + q_1^2}, \quad h_2 = Rq_1,$$

а главные кривизны k_1 и k_2 поверхности, вычисляемые по формулам (16), равны:

$$k_1 = -\frac{1}{R(1 + q_1^2)^{\frac{3}{2}}}, \quad k_2 = -\frac{1}{R\sqrt{1 + q_1^2}}.$$

Второе из уравнений (18) определяет связь между переменными q_1 и θ :

$$\dot{\theta} = \frac{\dot{q}_1}{1 + q_1^2},$$

откуда следует, что

$$q_1 = \operatorname{tg} \theta. \quad (27)$$

Учитывая формулу (27), мы можем считать теперь, что выражения для $\rho(q_1)$ и $\zeta(q_1)$ переписываются в виде:

$$\sigma(\theta) = \rho(q_1)|_{q_1=\operatorname{tg} \theta} = R \operatorname{tg} \theta, \quad \tau(\theta) = \zeta(q_1)|_{q_1=\operatorname{tg} \theta} = -\frac{R}{2} \operatorname{tg}^2 \theta.$$

В результате получим следующие выражения для параметров Ламе h_1 и h_2 и главных кривизн k_1 и k_2 :

$$h_1 = \frac{R}{\cos^3 \theta}, \quad h_2 = R \operatorname{tg} \theta, \quad k_1 = -\frac{\cos^3 \theta}{R}, \quad k_2 = -\frac{\cos \theta}{R}.$$

Линейное дифференциальное уравнение второго порядка (26) может быть представлено в виде:

$$\frac{d}{d\theta} \left(\frac{dv}{d\theta} + \frac{v}{\sin \theta \cos \theta} \right) = \frac{J}{J + ma^2} \frac{\sin^2 \theta}{\cos^2 \theta} v. \quad (28)$$

Таким образом, задача о качении шара по абсолютно шероховатой поверхности, такой что геометрическое место центров шара является параболоидом вращения, сводится к интегрированию линейного дифференциального уравнения второго порядка (28). Сделаем в уравнении (28) замену независимой переменной по формуле $x = \cos^2 \theta$ и введем обозначение

$$\frac{J}{J + ma^2} = n^2 < 1.$$

Тогда уравнение (28) приводится к уравнению с рациональными коэффициентами:

$$\frac{d^2v}{dx^2} + \frac{1}{x-1} \frac{dv}{dx} - \frac{(n^2 x^2 + 2(1-n^2)x + n^2 - 1)}{4x^2(x-1)^2} v = 0. \quad (29)$$

Для того чтобы привести дифференциальное уравнение (29) к более простому виду, сделаем замену

$$y = v \sqrt{x-1}.$$

Тогда линейное дифференциальное уравнение второго порядка (29) перепишется следующим образом:

$$\frac{d^2y}{dx^2} = \frac{n^2 - 1}{4x^2} y. \quad (30)$$

Уравнение (30) имеет в точности вид, необходимый для того, чтобы применить к данному уравнению алгоритм Ковачича [3]. Применение к уравнению (30) алгоритма Ковачича [3] показывает, что общее решение данного уравнения может быть представлено в виде:

$$y = C_1 x^{\frac{1+n}{2}} + C_2 x^{\frac{1-n}{2}},$$

где C_1 и C_2 — произвольные постоянные. Таким образом, можно сделать вывод, что общее решение исходного уравнения (28) выражается через лиувиллевы функции.

Заключение. Нами была рассмотрена задача о качении тяжелого однородного шара по выпуклой поверхности, такой что при качении центр тяжести шара принадлежит заданной поверхности вращения. Было показано, что решение задачи сводится к интегрированию линейного дифференциального уравнения второго порядка (26). Для случая, когда при качении шара его центр тяжести движется по параболоиду вращения, уравнение (26) было записано в явном виде (28) и было установлено, что данное уравнение интегрируется в лиувиллевых функциях.

Аналогичное исследование может быть проведено и для других поверхностей вращения, в частности, для различных поверхностей второго порядка. Исследования подобного рода мы планируем провести в будущем.

Список литературы

1. Routh E. J. The Advanced Part of a Treatise on the Dynamics of a System of Rigid Bodies: Being Part II of a Treatise on the Whole Subject. Cambridge: Cambridge University Press, 2013.
2. Noether F. Über rollende Bewegung einer Kugel auf Rotationsflächen. Leipzig: Teubner, 1909.
3. Kovacic J. An algorithm for solving second order linear homogeneous differential equations // J. Symb. Comput. 1986. Vol. 2. Issue 1. P. 3–43.
4. Kaplansky I. An Introduction to Differential Algebra. Paris: Hermann, 1957.
5. Do Carmo M. P. Differential geometry of curves and surfaces. New Jersey: Prentice-Hall, Englewood Cliffs, 1976.



Кулешов Александр Сергеевич — кандидат физ.-мат. наук, доцент кафедры теоретической механики и мехатроники механико-математического факультета МГУ им. М. В. Ломоносова, тел.: +7 (903) 536-87-22, e-mail: kuleshov@mech.math.msu.su.

Ведущий специалист в области механики систем с дифференциальными связями и динамики тел, взаимодействующих с твердой поверхностью. Лауреат конкурса на соискание медалей РАН с премиями для молодых ученых России за лучшую научную работу (2006 г.).

Alexander S. Kuleshov — PhD, associate professor at department of mechanics and mathematics, Lomonosov Moscow State University, e-mail: kuleshov@mech.math.msu.su, phone: +7 (903) 536-87-22.

Leading specialist in the field of mechanics of system with differential constraints and dynamics of a body, being contiguous to a rigid surface.



Соломина Дарья Владимировна — студентка 6-го курса кафедры теоретической механики и мехатроники механико-математического факультета МГУ им. М. В. Ломоносова, тел.: +7 (495) 939-36-81, e-mail: dasha.solomina@gmail.com.

Область научных интересов: динамика тел, взаимодействующих с твердой поверхностью.

Darya V. Solomina — 6th-year student at department of mechanics and mathematics, Lomonosov Moscow State University, phone: +7 (495) 939-36-81, e-mail: dasha.solomina@gmail.com.

Research interests: dynamics of a body, being contiguous to a rigid surface.



APPLICATION OF MACHINE LEARNING METHODS IN BUILDING CONSTRUCTION

T. V. Kupriyanova, D. I. Kislytsyn

Nizhny Novgorod State University of Architecture and Civil Engineering,
603950, Nizhny Novgorod, Russia

DOI: 10.24411/2073-0667-2021-10003

Construction is one of the largest sectors of the economy, which meets the needs of all rapidly developing countries. However, this industry is the most traumatic. Throughout the construction process, incidents of varying severity occur on construction sites every day. Along with this, the demand for innovative projects in the construction sector is increasing every year, so the question arose about the use of machine learning methods in production on the construction site. After all, it is thanks to artificial intelligence and machine learning that you can minimize project risks, secure and optimize construction processes. The purpose of the research work is to offer an option for the development of an automated module for construction production using machine learning methods. It provides functions for predicting and ranking incidents on the construction site, as well as automatically assigning a person responsible for closing the incident. To achieve this goal, certain tasks were solved. Namely: familiarization with existing methods of solving the problem, identification of possible options for collecting information about the situation at the construction site, analysis of available data for the distribution of incidents by their characteristics into appropriate categories, research of predictive machine learning algorithms and their ready-made implementations, and as a result, an automated module was presented at the initial stage of development and its effectiveness was evaluated. In the course of this work, a possible program interface with the necessary sections was demonstrated: the authorization (registration) page, the main page, the personal account, the sections of construction projects and providing data on incidents, as well as the incident archive and technical support. When the program is running, it communicates with the database, which will store all the data obtained for the entire period of construction production. In addition, the analysis of the received information from various sources of the case monitoring system in the construction industry will be carried out, then the incidents will be divided by signs into the appropriate categories and their color marking according to the degree of importance. In this case, each incident is recorded in a database that displays a set of single events that carry certain attributes. After analyzing an array of historical data, machine learning creates an algorithm for ranking incidents. The description of the incident is characterized by its type, time and place of occurrence, the device on which everything was observed, and some other information. All properties, with the exception of time, are divided into categories, and color-coded to facilitate recognition, depending on the degree of importance. For example, the 4th level is red — the most important, the 0th level is green—the problem is solved. The program also assumes the function of automatically creating requests for the elimination of incidents that have occurred and assigning responsible persons for closing emerging problem cases. Initially, machine learning receives data about employees who can eliminate the incident, and then, based on the standard machine learning algorithm — an ensemble of decision trees — the most suitable person is automatically selected. As a result of applying machine learning methods, the person responsible for solving the incident will be installed, to which all attributes obtained from monitoring systems are transmitted. And after the

problem is resolved, the responsible person changes the incident status to „Closed“. After that, the data becomes historical, and in the future can be used by machine learning when a similar situation occurs to guide its solution. When implementing the proposed automated module for construction production using machine learning methods, positive results are expected, including minimizing the number of missed incidents and reducing the response time to emerging problem situations.

Key words: machine learning algorithms, data mining, ranking of incidents on the construction site.

References

1. The rating of accidents at the construction site / [Electron Res.]: // Proraboff.RF website — https://xn--80ac1bcbgb9aa.xn--p1ai/rejting_neschastnyh_sluchaev_na_strojke/. Date of request 26.09.2020.
2. Hebert, Jeff. Predicting rare failure events using classification trees on large scale manufacturing data with complex interactions // 2016 IEEE International Conference on Big Data, BigData 2016, Washington DC, USA, December 5–8, 2016. P. 2024–2028.
3. Hastie, Trevor, Tibshirani, Robert, & Friedman, Jerome. The Elements of Statistical Learning. Springer Series in Statistics. New York, NY, USA: Springer New York Inc. 2001.
4. Smart construction technology: as machine learning to predict and prevent accidents on the construction site / J. Kanner. Your window to the world CADS: [Electron Res.]: http://isicad.ru/ru/articles.php?article_num=20712. Date of request 27.09.2020.
5. Application of machine learning to the ranking of incidents on the Moscow Railway / P. Y. Boyko, E. M. Bykov, E. I. Sokolov, D. A. Yarotsky. // Information technologies and computer systems. 2017. N 2. S. 43–53.



ПРИМЕНЕНИЕ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ В СТРОИТЕЛЬСТВЕ

Т. В. Куприянова, Д. И. Кислицын

Нижегородский государственный архитектурно-строительный университет
603950, Нижний Новгород, Россия

УДК 004.85, 69.05

DOI: 10.24411/2073-0667-2021-10003

Поскольку в строительной сфере ежегодно увеличивается спрос на инновационные проекты, возник вопрос о применении методов машинного обучения при производстве на строительной площадке. В данной статье описывается суть предлагаемого варианта разработки автоматизированного модуля для строительного производства с использованием методов машинного обучения. В нем представлены функции предсказания и ранжирования инцидентов на строительной площадке, а также автоматического назначения ответственного лица за закрытие инцидента. Продемонстрирован возможный интерфейс программы, в которой будет выполняться анализ поступившей информации с различных источников системы мониторинга случаев на строительном производстве, распределение инцидентов по признакам на соответствующие категории и их цветовая маркировка по степени важности, создание заявок на устранение возникших инцидентов и назначение ответственных лиц за закрытие возникающих проблемных случаев. При реализации предлагаемого автоматизированного модуля ожидаются положительные результаты, в числе которых минимизация количества пропущенных инцидентов и сокращение времени реакции на возникающие проблемные ситуации.

Ключевые слова: алгоритмы машинного обучения, интеллектуальный анализ данных, ранжирование инцидентов на строительной площадке.

Введение. Строительство — одна из крупнейших отраслей экономики, которая обеспечивает потребности всех быстроразвивающихся стран. Однако эта отрасль является самой травмоопасной. На протяжении всего строительного производства на строительных площадках каждый день происходят инциденты разной степени тяжести. По данным Роструда, 21 % работников от общего числа занятых на стройке стали участниками тяжелых несчастных случаев, большая часть которых закончилась летальным исходом [1]. Инцидент представляет собой совокупность ситуаций, которые обнаруживаются либо рабочими на стройплощадке, либо регистрируемыми датчиками и приборами и несущими признаки предотказного состояния и требующими технического обслуживания и ремонта.

Ежегодно в строительной сфере увеличивается спрос на инновационные проекты. Поэтому возник вопрос о применении методов машинного обучения при производстве на строительной площадке. Ведь именно благодаря искусственному интеллекту и машинному обучению можно минимизировать проектные риски, обезопасить и оптимизировать строительные процессы.

Научной проблемой, на решение которой направлено исследование, является недостаточная автоматизация технологических процессов для предотвращения и устранения инцидентов на строительных объектах.

Целью проводимой научно-исследовательской работы является предложение варианта разработки автоматизированного модуля для строительного производства с использованием методов машинного обучения с функциями предсказания и ранжирования инцидентов на строительной площадке, автоматического назначения ответственного, которые минимизируют количество пропущенных инцидентов и сократят время реакции на возникающие проблемные ситуации. Для достижения поставленной цели планируется решение следующих задач:

- 1) ознакомление с существующими способами решения проблемы,
- 2) выявление возможных вариантов сбора информации о ситуации на строительном объекте,
- 3) проведение анализа имеющихся данных для распределения инцидентов по их признакам на соответствующие категории,
- 4) исследование предсказательных алгоритмов машинного обучения и их готовых реализаций,
- 5) представление автоматизированного модуля на начальном этапе разработки и оценка его эффективности.

В настоящее время машинное обучение широко применяется во многих сферах [2, 3] для решения задач предсказательного обслуживания промышленного сегмента. Однако использование методов машинного обучения в строительной отрасли происходит на менее продвинутом уровне. Это в большей степени технологии для определения и анализа угроз, например, для безопасности на стройке. Примером такой разработки является стартап Smartvid.io, разработавший технологию применения искусственного интеллекта и машинного обучения, которая анализирует фото- и видеинформацию на стройплощадке [4]. Эта технология дает пользователям возможность получить данные и изображения с площадок для обеспечения безопасности на объекте.

Использование машинного обучения в качестве решения задачи предсказательного обслуживания произошло после доработки проекта Smartvid.io, где в результате сотрудничества с компанией Suffolk Construction была разработана возможность предсказывания инцидентов на конкретных строительных площадках. Для этого использовались сохраненные за десять лет данные о строительных процессах от Suffolk. Они были представлены в движке на базе искусственного интеллекта — Vinnie. Во время процесса работы происходило сравнение ситуаций в настоящее время с моментами из информационной базы [4]. На основе исторических данных могли предсказываться возможные инциденты и выдавались предупреждения.

Данные проекты действительно эффективные, однако они неполные, поскольку вся информация, которая обрабатывается, не структурирована, выводится сплошным потоком и не имеет рекомендаций ни по предотвращению или устранению инцидента, ни по ответственным лицам, которые должны решить появившуюся проблему. Именно поэтому необходимо создать автоматизированный модуль на основе машинного обучения, который, помимо ведения непрерывного мониторинга инцидентов, определения и анализа угроз, будет составлять приоритетный список проблем, на которые нужно обратить внимание в первую очередь, а также классифицировать, маркировать их на основе визуальных данных и данных с датчиков и определять ответственного за устранение возможной или существующей проблемы.

Для сбора информации о ситуации на строительном объекте необходимо использовать все известные на текущий момент параметры инцидента (фото- и/или видеоматериалы,

признаки места, времени, этап строительства, показания датчиков и приборов и т. п.), а также его контекст (погода, прошлые инциденты и др.). Помимо этого, существуют функции системы мониторинга: контроль над техническим состоянием объектов в реальном масштабе времени; диагностика и прогнозирование состояния объектов; определение предотказных состояний объектов.

Нередко во время строительных работ возникает множество различных проблем разной степени сложности. Случай автоматически объединяются в инциденты в информационной системе, а затем текущие инциденты обрабатываются людьми, которые отвечают за мониторинг инцидентов. Эти работники выявляют причины возникновения неполадок, классифицируют как относящиеся к одному из нескольких типов (предотказному состоянию, техническому обслуживанию и ремонту, недостатку диагностики или технологической ситуации) и принимают необходимые меры. Обработка всех этих операций зачастую может быть крайне интенсивной и трудоемкой: новые инциденты могут появляться с промежутком в несколько секунд; в то время как машинное обучение может справиться с этой задачей гораздо быстрее и эффективнее. Именно поэтому крайне актуально использовать автоматическое ранжирование инцидентов по уровню их важности.

Важность инцидента можно определить двумя факторами: риском реального предотказного состояния инцидента и последствиями (в частности, экономическим эффектом) отказа устройства, использующегося на строительном объекте. Однако алгоритмы машинного обучения следует использовать только для предсказания риска предотказного состояния на основе автоматически сформированных данных, полученных от приборов и датчиков, об инциденте, поскольку эти факторы разные по характеру, и оценка последствий отказа предполагает дополнительные стоимостные модели отказов [5]. В зависимости от важности инцидента машинным обучением будут указаны баллы от 0 до 4 между ситуациями, где 4 — наиболее важно, 0 — низкая степень приоритета.

Каждый инцидент заносится в базу данных, которая отображает набор единичных событий, несущих определенные атрибуты. Все эти ситуации автоматически объединяются в инциденты на этапе предварительной обработки сигналов датчиков. После анализа массива исторических данных, машинное обучение создает алгоритм ранжирования инцидентов. Описание инцидента характеризуется своим типом, временем и местом проявления, прибором, на котором все наблюдалось, и некоторыми другими сведениями. Все свойства, за исключением времени, распределяются на категории. В соответствии с определенным набором характеристик конкретного инцидента следует присвоить ему одну из пяти категорий с цветовой маркировкой для облегчения распознавания: четвертый уровень (красный) — ситуация, требующая срочного устранения проблемы и несущая за собой жертвы; третий уровень (оранжевый) — ситуация, требующая внимание, и которая может привести к возможным жертвам; второй уровень (желтый) — небольшой сбой, не приводящий к возможным жертвам; первый уровень (серый) — несрочная и незначительная проблема; нулевой уровень (зеленый) — проблема решена.

При появлении нового инцидента и его автоматическом ранжировании следующим шагом является назначение ответственного за решение проблемы. Для этого необходим анализ статистики по всем работникам на стройке, когда-либо участвовавшим в исправлении неполадок во время строительного процесса, среди которых выбираются те, кто работает в текущее время и не является ответственным за неисправность третьего или четвертого уровня приоритета. Практика машинного обучения производит построение модели: сначала отделяются атрибуты и создается обучающая матрица, а затем применяется некоторый

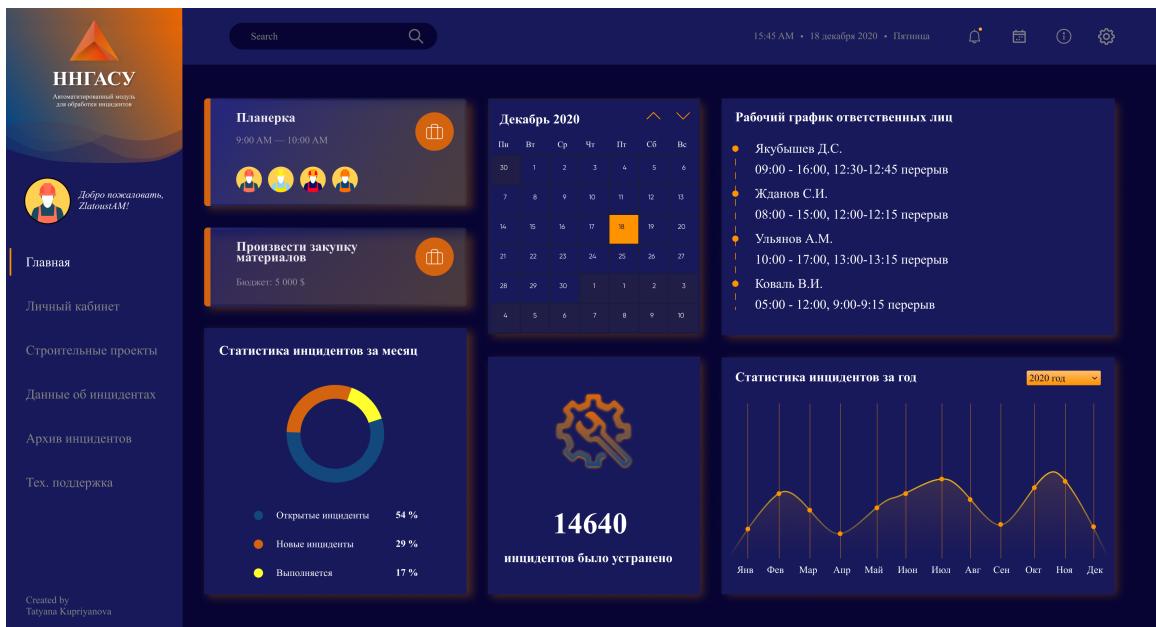


Рис. 1. Вид главной страницы программы автоматизированного модуля для обработки инцидентов

общий алгоритм машинного обучения к полученной обучающей матрице. Важным шагом для подготовки данных является способ их преобразования в числовые вектора определенной размерности, атрибуты которого отражают существенные характеристики работника. Вся выборка передается некоторому классификатору, основанному на стандартном алгоритме машинного обучения — ансамбле решающих деревьев. И в результате применения методов машинного обучения будет установлен ответственный за решение инцидента, которому передаются все атрибуты, полученные из систем мониторинга.

В конечном итоге, после решения и закрытия проблемы, данные становятся историческими и в дальнейшем могут быть использованы машинным обучением при появлении подобной ситуации для руководства по ее решению.

На основе всех вышеописанных технологий можно разработать автоматизированный модуль для обработки инцидентов на строительной площадке с использованием методов машинного обучения.

Предлагаемая разработка автоматизированного модуля для обработки инцидентов представляет из себя программу, которая связана с базой данных, в которой будет храниться вся информация об инцидентах, а также которая принимает показания с устройств системы мониторинга.

На главной странице показываются возможные объявления, автоматически загруженные с сайта организации, краткая общая статистика по текущему месяцу в виде круговой диаграммы и за весь год в виде графика, а также календарь и рабочий график всех ответственных лиц, которые могут заниматься устранением инцидентов. Вид главной страницы программы автоматизированного модуля для обработки инцидентов представлен на рис. 1.

Для входа в программу ответственное лицо авторизуется в личном кабинете. Если же это новый работник, ему необходимо зарегистрироваться в системе. На рис. 2 представлены формы авторизации и регистрации программы.

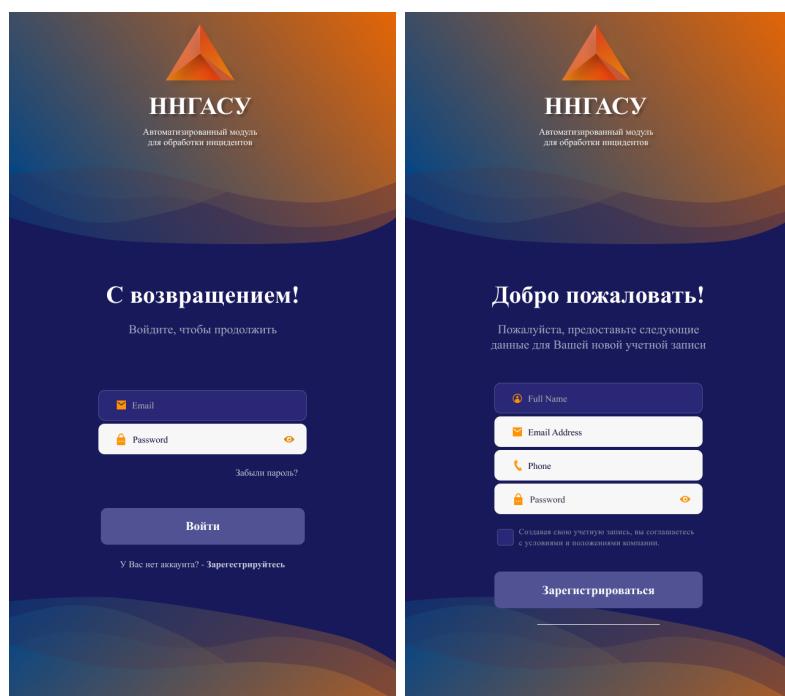


Рис. 2. Вид формы авторизации и регистрации программы автоматизированного модуля для обработки инцидентов

Помимо основных данных, после регистрации нужно будет указать дополнительную информацию: должность, квалификацию, график работы. Полученные данные необходимы для контактной базы данных всех работников организации и становятся доступными для обработки машинным обучением. Раздел „Личный кабинет“ показывает профиль сотрудника, под логином которого прошла авторизация, его фотографию, график работы, а также историю текущих заявок и тех, которые он когда-либо ликвидировал. Здесь же работник может посмотреть дополнительную информацию о заявке и добавить комментарии к инциденту, а также закрыть его после устранения проблемы. В личном кабинете можно редактировать информацию, перейти в раздел сообщений, открыть список проектов или других рабочих, выставить пользовательские настройки. На рис. 3 представлен вид раздела личного кабинета со стороны ответственного лица.

Поскольку в организации в процессе работы может быть одновременно несколько строительных проектов, в настройках программы или в разделе „Строительные проекты“ выбирается среди выпадающего списка конкретный объект, информация по которому в дальнейшем будет отображаться в разделе „Данные об инцидентах“. В противном случае, если изначально объект не будет выбран, в категории, где отображаются данные об инцидентах, будет собрана информация по всем строительным проектам. После выбора объекта, по нему отображается следующие сведения: дата начала работ на строительной площадке, информация о заказчике, стройгенплан и главные задачи, которые необходимо выполнить.

Раздел „Данные об инцидентах“ является ключевым и содержит краткие сведения о возникших инцидентах. С помощью машинного обучения автоматически выводится и обновляется информация об инцидентах определенного строительного проекта: публикуются актуальные данные об открытых и новых (необработанных) инцидентах, а также их количество на стадии устранения. Данные в таблице заполняются автоматически машинным

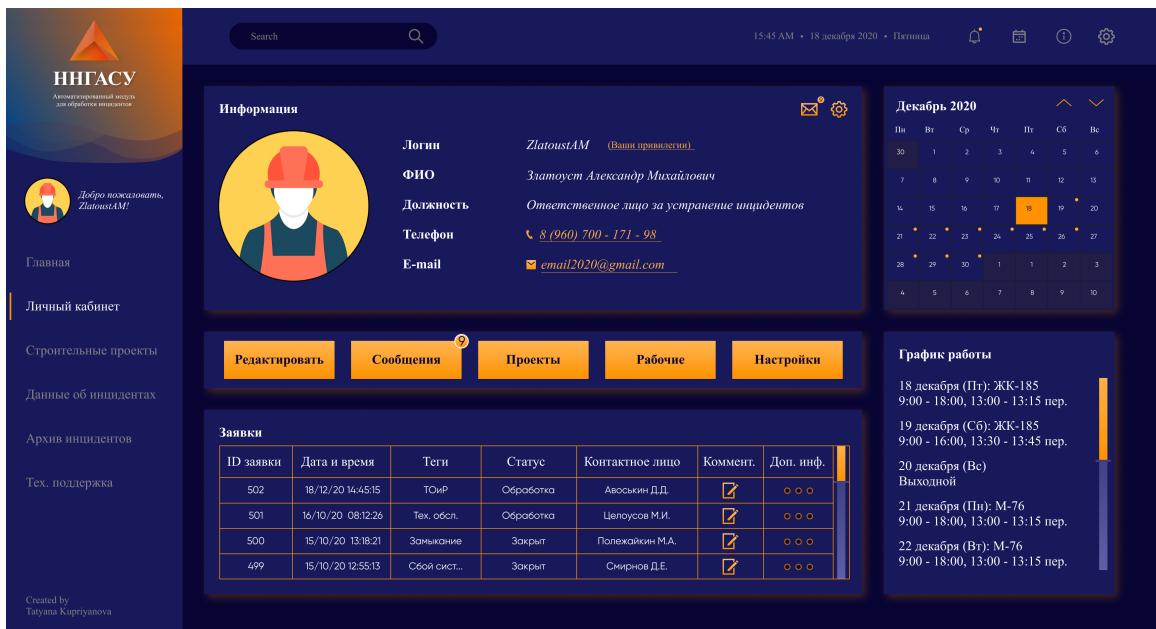


Рис. 3. Вид раздела „Личный кабинет“ со стороны ответственного лица программы автоматизированного модуля для обработки инцидентов

обучением на основе показаний с датчиков, приборов и других устройств мониторинга: ID инцидента; дата и время его возникновения; теги, которые выводятся машинным обучением в ходе анализа имеющихся данных о ситуации; статус инцидента; ответственный за ликвидацию инцидента; дополнительная информация; приоритет, маркированный цветовой гаммой для облегчения восприятия информации. Если возникает спорная ситуация, когда существуют инциденты с одинаковым приоритетом, то с помощью алгоритма машинного обучения происходит сравнение инцидентов по их дате и времени возникновения. Наибольший приоритет отдается той заявке, которая создана раньше. Помимо этого, раздел с данными об инцидентах имеет функцию поиска информации по ключевым словам, а также фильтрации открытых инцидентов по трем критериям: дате возникновения, тегам и лицу, ответственному за инцидент. Вид страницы, содержащей данные об инцидентах, программы автоматизированного модуля для обработки инцидентов представлен на рис. 4.

При нажатии на кнопку с многоточием в столбце „Дополнительная информация“ открывается новое окно с развернутыми характеристиками, помимо уже показанных кратких сведений, указываются источники информации (код и наименование устройства мониторинга, с которого пришел сигнал); местоположение возникшего инцидента с координатами; время сбоя связи с тем устройством, с которым произошел инцидент; перечисляются предыдущие поломки, если они имелись; показывается статистика уровня сигнала связи с тем устройством, с которым произошел инцидент; прикрепляются доступные медиафайлы, которые были получены от устройств мониторинга в виде фото и/или видео, с возможностью ручного добавления файлов; доступно поле ввода для комментария ответственного за инцидент, который после устранения инцидента производит изменение статуса инцидента на „Закрыт“, то есть приоритет выставляется нулевой. Все данные также сохраняются в базе и автоматически сохраняются при изменении. Вид окна с дополнительной информацией по инциденту представлен на рис. 5.

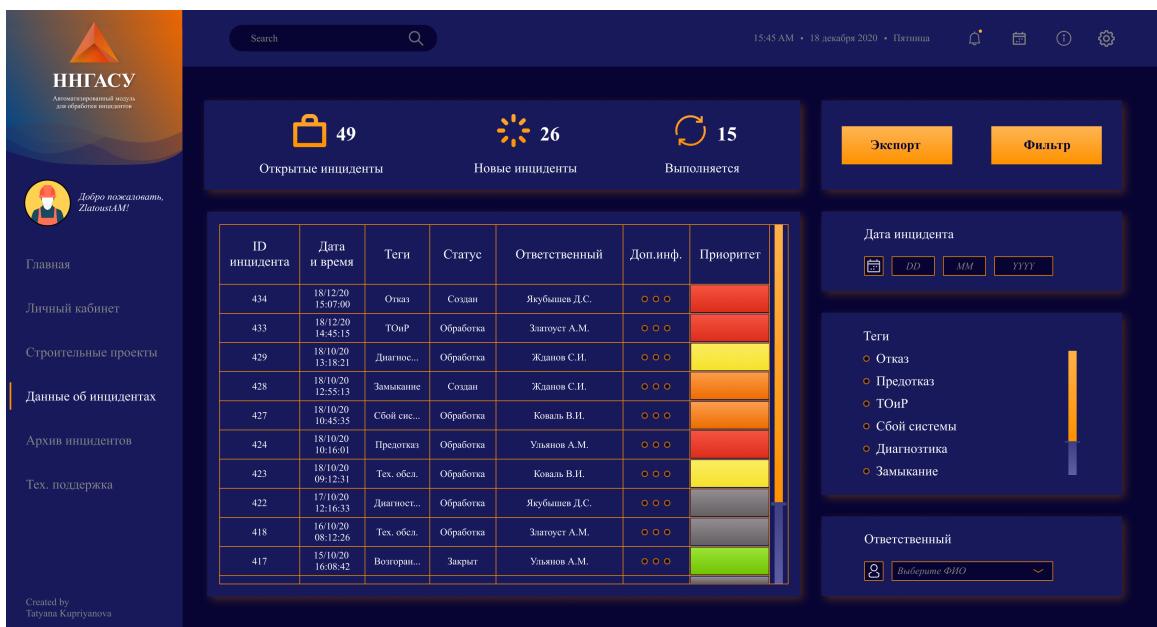


Рис. 4. Вид раздела „Данные об инцидентах“ программы автоматизированного модуля для обработки инцидентов

В архиве инцидентов содержится вся краткая и дополнительная историческая информация об инцидентах с аналогичными возможностями поиска по ключевым словам и фильтрации инцидентов на основе тех же параметров, как в разделе текущих инцидентов, а также добавочным критерием является поиск по конкретным строительным проектам.

Раздел технической поддержки необходим для поддержания работоспособности программы и при необходимости ее доработки или устранения возможных ошибок. Для обычных пользователей в нем содержится форма заявки с указанием типа неисправности, описанием возникшей проблемы с возможностью прикрепления скриншотов. У администратора, который устраняет возможные ошибки программы, автоматически после обработки информации появляется заявка, которую можно увидеть в разделе технической поддержки или в личном кабинете в поле „Заявки“. Заполнение информационной таблицы происходит машинным обучением по той же технологии, что и с инцидентами. Различие лишь в том, что в данном случае информация вводится вручную другими сотрудниками организации.

Подобная разработка программы автоматизированного модуля для обработки инцидентов с использованием алгоритмов машинного обучения будет эффективной, поскольку позволяет сократить количество пропущенных инцидентов, уменьшить расходы и время реакции на возникшую проблемную ситуацию, что положительно отразится на всех процессах строительного объекта.

В данной работе был предложен вариант разработки автоматизированного модуля для строительного производства, основанного на методах машинного обучения, в ходе которых были рассмотрены задачи автоматического ранжирования инцидентов и назначения ответственных лиц за их ликвидацию. При реализации этого автоматизированного модуля можно ожидать положительные практические результаты.

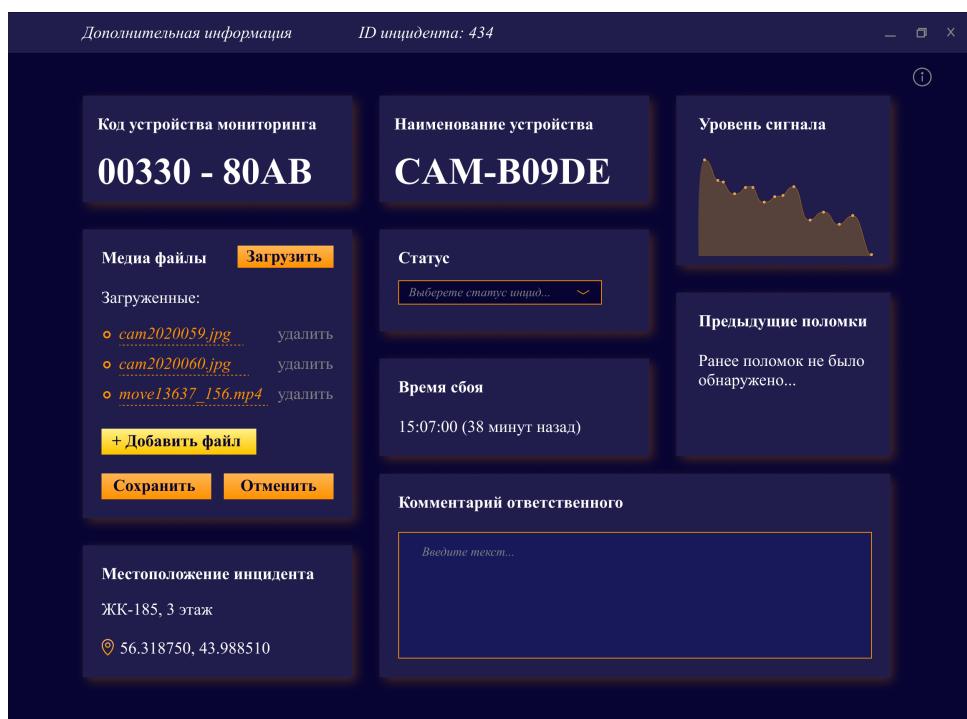


Рис. 5. Вид окна с дополнительной информацией по инциденту в программе автоматизированного модуля для обработки инцидентов

Список литературы

1. Рейтинг несчастных случаев на стройке / Прорабофф.РФ. [Электрон. рес.]: https://xn--80ac1bcgb9aa.xn--p1ai/rejting_neschastnyh_sluchaev_na_strojke/. Дата обращения 26.09.2020.
2. Hebert, Jeff. Predicting rare failure events using classification trees on large scale manufacturing data with complex interactions. // 2016 IEEE International Conference on Big Data, BigData 2016, Washington DC, USA, December 5–8, 2016. P. 2024–2028.
3. Hastie, Trevor, Tibshirani, Robert, & Friedman, Jerome. The Elements of Statistical Learning. Springer Series in Statistics. New York, NY, USA: Springer New York Inc. 2001.
4. Технология умного строительства: как машинное обучение предсказывает и предотвращает несчастные случаи на стройке / Дж. Каннер. Ваше окно в мир САПР. [Электрон. рес.]: http://isicad.ru/articles.php?article_num=20712. Дата обращения 27.09.2020.
5. Применение машинного обучения к ранжированию инцидентов на Московской железной дороге / П. Ю. Бойко, Е. М. Быков, Е. И. Соколов, Д. А. Яроцкий. // Информационные технологии и вычислительные системы. 2017. № 2. С. 43–53.

Куприянова Татьяна Викторовна — студентка третьего курса бакалавриата по направлению „Информационные системы и технологии“ Нижегородского государственного архитектурно-строительного университета. E-mail: kupryanova.tanya@vandex.ru. Имеет две публикации в научном журнале „Системный администратор“, участница таких научных кон-

ференций, как X Всероссийский Фестиваль науки, 20-я международная конференция „Математическое моделирование и суперкомпьютерные технологии“. Область научных интересов: искусственный интеллект, машинное обучение, интернет вещей (IoT), технологии виртуальной и альтернативной реальности.



Kupriyanova Tatyana Viktorovna is a third-year bachelor's student in the field of „Information Systems and Technologies“ of the Nizhny Novgorod University of Architecture and Civil Engineering. E-mail: kuprvanovatanya@vandex.ru.

She has two publications in the scientific journal „System Administrator“, participated in such scientific conferences as the X All-Russian Science Festival, the 20th International Conference „Mathematical Modeling and Supercomputer Technologies“. Research interests: artificial intelligence, machine learning, Internet of Things (IoT), virtual and alternative reality technologies.

Кислицын Дмитрий Игоревич — канд. технич. наук, доцент кафедры информационных систем и технологий Нижегородского государственного архитектурно-строительного университета. E-mail: kislitsynd@yandex.ru.

В 2004 году окончил ПГС ННГАСУ, в 2009 году защитил диссертацию на соискание степени кандидата технических наук по специальности 05.13.12 „Системы автоматизации проекти-

рования“, в 2013 году присвоено ученое звание доцента на кафедре информационных систем и технологий. Имеет 46 научных публикаций и свидетельство о государственной регистрации программы для ЭВМ. Область научных интересов: разработка автоматизированных систем конструкторского расчета сложных строительных конструкций, компьютерное моделирование карстовых провалов, интернет вещей (IoT), машинное обучение.



Kislitsyn Dmitriy Igorevich — graduated from Nizhny Novgorod University of Architecture and Civil Engineering in 2004. Ph.D. in Technical Sciences in 2009. Docent in 2013.

Associate Professor of the Department of Information Systems and Technologies, Nizhny Novgorod State University of Architecture and Civil Engineering. E-mail: kislitsynd@yandex.ru.

Research interests: development of automated systems for structural analysis of complex building constructions, computer modeling of karst sinkholes, Internet of Things (IoT), machine learning.

Дата поступления — 03.02.2021

FINITE ELEMENT MODEL OF SEA ICE DYNAMICS AND ITS PARALLEL IMPLEMENTATION USING THE INMOST LIBRARY

S. S. Petrov

Marchuk Institute of Numerical Mathematics of the Russian Academy of Sciences (INM RAS),
119333, Moscow, Russia

Moscow Center for Fundamental and Applied Mathematics at the INM RAS,
Moscow, Russia

DOI: 10.24411/2073-0667-2021-10004

This work is devoted to the numerical implementation of the dynamic core of finite-element sea ice drift model with nonlinear viscous-plastic rheology. Sea ice models of this level are an integral part of modern ocean dynamics models, which are used both for short-term and seasonal weather forecasts and for predicting changes in the Earth's climate. A technology for generating an irregular triangulation for the Arctic Ocean and adjacent seas with a thickening of the grid in areas with a potentially high ice concentration, near the coast, and in narrow straits, taking into account data on the contours of the continents and data of ice cover, is described. A description of the algorithm for interpolating geodata into model grid nodes is given. In the second part of the work, a finite element implementation of the model with various time integration schemes for the momentum balance equation is presented. To integrate the equations of mass and ice concentration transfer, the Taylor-Galerkin scheme with flux correction was implemented. An optimization of the time integration scheme for the momentum balance equation is proposed, which allows to accelerate the standard stationary mEVP method used in modern ice drift models (CICE, LIM, FESIM). The idea of the proposed nonstationary mEVP-opt method is to approximate the iteration parameter to the locally optimal one obtained from the estimation of the integration step in the approximation of the linearized transition operator. The results of a model numerical experiment to reproduce the most computationally complex mode of slow ridging are presented. The result is compared to the Picard method one with 10 pseudo-iterations, which gives high accuracy at high computational costs. It is shown that the new mEVP-opt method provides a significant reduction in the computation time compared to mEVP with a slight increase in the number of operations.

Key words: modeling of sea ice dynamics, viscous-plastic rheology, INMOST, Ani-2D, Ani-3D.

References

1. Hibler W. D. A Dynamic Thermodynamic Sea Ice Model // *Journal of Physical Oceanography*. 1979. V. 9. N 4. P. 815–846.
2. Hunke E. C., Dukowicz J. K. An Elastic-Viscous-Plastic Model for Sea Ice Dynamics // *Journal of Physical Oceanography*. 1997. V. 27. N 9. P. 1849–1867.
3. Bouillon S., Fichefet T., Legat V., Madec G. The elastic-viscous-plastic method revisited // *Ocean Modelling*. 2013. V. 71. P. 2–12.
4. Kimmritz M., Danilov S., Losch M. The adaptive EVP method for solving the sea ice momentum equation // *Ocean Modelling*. 2016., V. 101.

5. Wessel P., Smith W. H. F. A Global Self-consistent, Hierarchical, High-resolution Shoreline Database // *Journal of Geophysical Research*. 2000. V. 101, P. 8741–8743.
6. Wessel P., Luis J.F., Uieda L., Scharroo R., Wobbe F., Smith W. H. F., Tian D. The Generic Mapping Tools version 6 // *Geochemistry, Geophysics, Geosystems*. 2019. V. 20. P. 5556–5564.
7. Danilov A. Unstructured tetrahedral mesh generation technology // *Comp. Math. and Math. Phys.* 2010. V. 50. N 1. P. 139–156.
8. Meier W. N., Fetterer F., Windnagel A. K. 2017. Near-Real-Time NOAA/NSIDC Climate Data Record of Passive Microwave Sea Ice Concentration. V. 1.
9. Danilov A., Terekhov K., Konshin I., Vassilevski Y. INMOS parallel platform: Framework for numerical modeling // *Supercomputing Frontiers and Innovations*. 2015. V. 2. P. 55–66.
10. Karypis G., Kumar V. A Fast and Highly Quality Multilevel Scheme for Partitioning Irregular Graphs // *SIAM Journal on Scientific Computing*. 1999. V. 20. N 1. P. 359–392.
11. [Electron. Res.]: <https://www.mcs.anl.gov/petsc/>.
12. Sakov P., Counillon F., Bertino L., Liszter K. A., Oke P. R., Koralev A. TOPAZ4: an ocean-sea ice data assimilation system for the North Atlantic and Arctic // *Ocean Sci.* 2012. V. 8. P. 633–656.
13. [Electron. Res.]: <https://www.copernicus.eu/en>.
14. [Electron. Res.]: <https://www.unidata.ucar.edu/software/netcdf/docs/index.html>.
15. Hutchings J. K., Jasak H., Laxon S. W. A strength implicit correction scheme for the viscous-plastic sea ice model // *Ocean Modell.* 2004. V. 7. P. 111–133.
16. Kuzmin D., Turek S. Flux correction tools for finite elements // *J. Comput. Phys.* 2001. P. 525–558.
17. Olshanskij M. A. Lekciiuprazhneniya po mnogosetochnym metodam // *Fizmatlit*. 2005. S. 24–25.
18. Danilov S., Wang Q., Timmermann R., Iakovlev N., Sidorenko D., Kimmritz M., Jung T., Schröter J. Finite-Element Sea Ice Model (FESIM), version 2 // *Geosci. Model Dev.* 2015. V. 8. P. 1747–1761.



КОНЕЧНО-ЭЛЕМЕНТНАЯ МОДЕЛЬ ДИНАМИКИ МОРСКОГО ЛЬДА И ЕЕ ПАРАЛЛЕЛЬНАЯ РЕАЛИЗАЦИЯ С ИСПОЛЬЗОВАНИЕМ БИБЛИОТЕКИ INMOST

С. С. Петров

Институт Вычислительной Математики им. Г. И. Марчука Российской Академии Наук
(ИВМ РАН),
119333, Москва, Россия
Московский центр фундаментальной и прикладной математики в ИВМ РАН,
119333, Москва, Россия

УДК 004.942, 519.635.8

DOI: 10.24411/2073-0667-2021-10004

Данная работа посвящена описанию численной реализации динамического ядра конечно-элементной модели дрейфа морского льда с учетом нелинейной вязко-пластичной реологии. Модели морского льда такого уровня являются неотъемлемой частью современных моделей динамики океана, которые используются как для краткосрочных и сезонных прогнозов погоды, так и для предсказаний изменений климата Земли. Описывается технология построения нерегулярной триангуляции для области Северного Ледовитого океана и прилегающих морей со сгущением сетки в районах с потенциально высокой концентрацией льда, у берега и в узких проливах с учетом данных о контурах материков и данных о состоянии ледового покрова. Даётся описание алгоритма интерполяции геоданных в узлы модельной сетки. Во второй части работы представлена конечно-элементная реализация модели с различными схемами интегрирования по времени уравнения баланса импульса. Для интегрирования уравнений переноса массы и сплошности льда реализована схема Тейлора-Галеркина с коррекцией потоков. Предложена оптимизация схемы интегрирования по времени уравнения баланса импульса, позволяющая ускорить стандартный стационарный mEVP-метод, применяемый в современных моделях дрейфа льда (CICE, LIM, FESIM). Идея предлагаемого нестационарного метода mEVP-opt состоит в приближении параметра итерации к локально-оптимальному, полученному из оценки шага интегрирования в приближении линеаризованного оператора перехода. Представлены результаты модельного численного эксперимента по воспроизведению наиболее сложного с вычислительной точки зрения режима медленного торошения. Результаты расчетов сравниваются с результатами метода Пикара с 10 псевдоитерациями, который дает высокую точность при больших вычислительных затратах. Показано, что новый метод mEVP-opt дает существенное уменьшение времени счета по сравнению с mEVP при незначительном увеличении числа арифметических операций.

Ключевые слова: моделирование динамики морского льда, вязко-пластичная реология, INMOST, Ani-2D, Ani-3D.

Работа выполнена в Институте вычислительной математики им. Г. И. Марчука РАН при финансовой поддержке Московского центра фундаментальной и прикладной математики (соглашение с Минобрнауки России № 075-15-2019-1624). Автор выражает благодарность сотрудникам ИВМ РАН: Н. Г. Яковлеву за научное руководство, П. А. Пережогину за ценные советы, А. А. Данилову за помощь с построением сеток, а также В. К. Крамаренко за помощь с освоением программного комплекса INMOST.

Введение. Модели льда играют важную роль в прогностических системах и климатических моделях. Наличие льда существенно влияет на тепломассообмен океана с атмосферой. Практический интерес для проектирования сооружений добычи шельфовых углеводородов и для прогнозирования маршрутов Арктических экспедиций представляет непосредственно динамика льда.

Представленная работа состоит из двух частей. Первая часть посвящена построению триангуляции расчетной области — области Северного Ледовитого океана со всеми прилегающими морями и детальным описанием линии берега. Далее следует описание параллельной системы интерполяции геоданных на модельную сетку, которая применяется для насыщения разрабатываемой модели данными, и формирования вывода. В основном, геоданные в мировом научном сообществе принято распространять в формате netCDF на прямоугольной сетке, возможно, с уровнями по глубине. Поскольку разрабатываемая модель строится на неструктурированной треугольной сетке, процесс интерполяции скаляров и векторов с прямоугольной на треугольную сетку и обратно требует особого пояснения.

Вторая часть работы посвящена описанию оптимизации самого распространенного метода численного интегрирования уравнения баланса импульса. Активное развитие моделей льда началось после разработки уравнений динамики морского льда с вязко-пластичной реологией [1]. Впоследствии было предложено множество методов численного интегрирования уравнений баланса импульса [2, 3] и их оптимизаций [4]. В данной работе представлен нестационарный метод, основанный на классическом mEVP-подходе [3] и идее локального убывания квадрата нормы невязки линеаризованного функционала, увеличивающей скорость сходимости. Данный метод может быть применен для оптимизации вычислений в блоках динамики морского льда. Главным достоинством конечно-элементного подхода для моделирования морского льда является возможность точного учета неоднородности береговой линии, возможность сгущения сетки в области особенностей решения и районы, представляющие интерес для пользователя. Физическая постановка данной задачи приводит к плохо обусловленным дискретным операторам, что сказывается на вычислительной сложности решения, поэтому оптимизации методов типа mEVP нуждаются в разработке для эффективного и экономичного использования моделей льда в связке с моделями океана.

1. Построение модельной сетки. В качестве расчетной области выбран Арктический регион, находящийся выше 45 широты. Контуры берегов взяты из открытой базы данных GSHHG [5] береговой линии, с помощью пакета GMT [6]. В качестве отправной точки используется самое грубое доступное разрешение. Несмотря на это, как показала практика, в представленных контурах берега присутствуют несогласованности, выражющиеся в самопересечении отрезков границы берегов. Еще одним недостатком является присутствие узких заливов и смежных отрезков берега, образующих слишком острые углы. Для устранения этого недостатка применяется следующая методика сглаживания, которая преследует основную цель: можно удалять некоторые вершины в ломанной береговой линии, но нельзя их двигать. Данное требование является необходимым, во-первых, для написания автоматизированной программы сглаживания береговой линии для произвольного „плохого“ контура, во-вторых, оставляет возможность добавления более детальной части ломанной берега из той же базы данных между существующими вершинами. Выполняется процедура с параметрами:

- 1) Если длина отрезка превышает заданную величину, то его конечная точка удаляется;
- 2) Если два смежных отрезка образуют угол, меньший наперед заданной величины, то

вершина угла удаляется; 3) Зафиксируем вершину и натуральное число n . Если среди n следующих отрезков есть отрезок, приближенный к выбранной точке на расстояние, меньшее фиксированной величины, то этот отрезок и все предыдущие точки (вплоть до фиксированной точки) удаляются; 4) Каждый остров, состоящий из менее 5 отрезков, удаляется.

Также на начальном этапе построения сетки были удалены устья рек, а расчетная область замыкалась по водной границе, проходящей вдоль 45 широты. После описанных модификаций, данные ломанных береговой линии подавались на вход процедуре построения регулярной триангуляции пакета Ani-2D AFT [7], разработанного в ИВМ РАН. Результат регулярной триангуляции представлен на рис. 1.

Регулярная триангуляция имеет явный недостаток — в низких широтах льда практически не бывает, избыточное количество узлов в таких местах приводит к лишним вычислениям. Поэтому была применена процедура сгущения сетки в области с потенциально высокой сплоченностью льда. Для этого использовались данные по спутниковым измерениям за последние 10 лет [8] и специальный функционал пакета Ani-2D AFT, позволяющий задать желаемый локальный размер треугольника в конкретной точке. Граница льда с высокой и низкой сплоченностью является довольно резкой. Для того чтобы сгладить границу, спутниковые данные по сплоченности льда в узлах прямоугольной сетки преобразовывались с помощью дискретного оператора Лапласа

$$a_{i,j}^{\text{new}} = -\frac{a_{i-1,j} + a_{i+1,j} + a_{i,j-1} + a_{i,j+1} - 4a_{i,j}}{h^2},$$

где h — шаг двумерной сетки. Минимальная возможная концентрация считается равной $a_{\min} = 0.05$. Максимальный размер треугольника задавался в 5 раз больше минимального $d_{\max} = 5d_{\min}$. Если $a(x, y)$ — значение билинейного интерполянта сплоченности в точке (x, y) , то желаемый размер треугольника в этой точке $d(x, y)$ вычисляется по формуле

$$d(x, y) = d_{\max} + \frac{d_{\min} - d_{\max}}{1 - a_{\min}} \cdot (a(x, y) - a_{\min}).$$

Результат триангуляции со сгущением сетки в область с высокой потенциальной сплоченностью льда для $d_{\min} \approx 10$ км. представлен на рис. 2.

Построенная модельная область состоит из 329 670 узлов, 642 387 треугольников и 17 049 граничных ребер. Понятно, что решения задач подобных размерностей нужно проектировать, учитывая параллельную архитектуру суперкомпьютера.

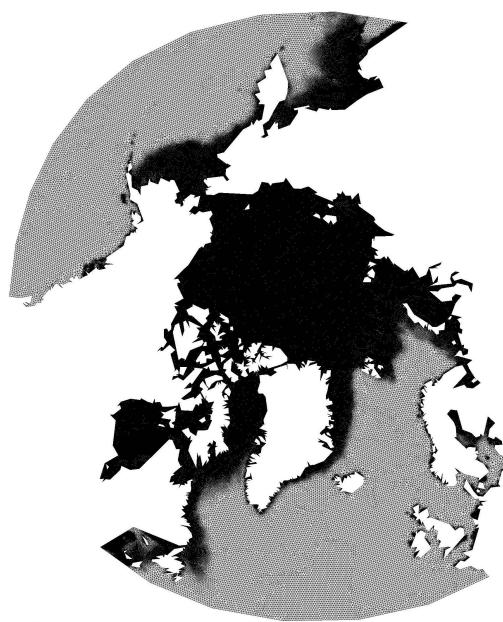
Стоит отметить, что использование географических (долгота/широта) координат в качестве модельных затруднительно в связи с особенностью на Северном полюсе. Для обхода этой проблемы модельные координаты отличаются от географических поворотом Северного полюса на географический экватор. Это достигается поворотом на углы Эйлера $(-30^\circ, -90^\circ, 0^\circ)$.

2. Особенности использования пакета INMOST для массивно-параллельного моделирования динамики морского льда. Программный комплекс INMOST [9] разрабатывается и поддерживается в ИВМ РАН. Данный пакет предназначен для массивно-параллельного моделирования на сетках произвольной структуры, как в двухмерном, так и в трехмерном пространстве. INMOST хорошо оптимизирован для использования конечно-элементных и конечно-объемных аппроксимаций. Пакет INMOST написан на языке программирования C++ и включает в себя функционал MPI и OpenMP для многопроцессорных и многопоточных вычислений. Для построения эффективного параллельного



mesh

Рис. 1. Регулярная триангуляция



mesh

Рис. 2. Триангуляция со сгущением

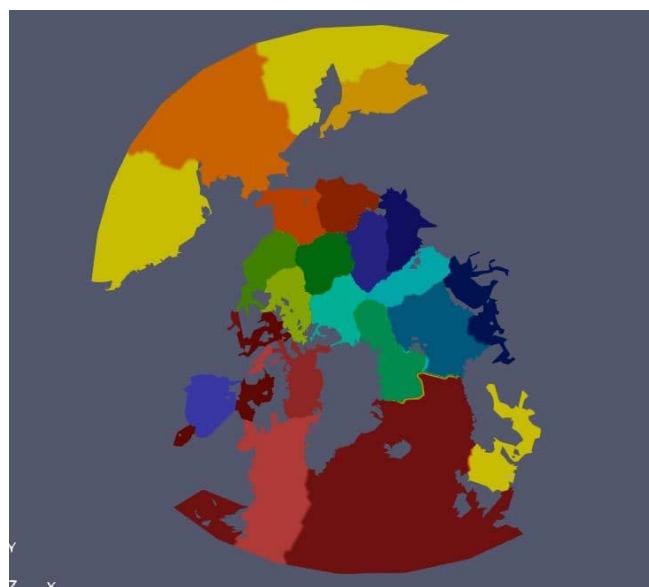


Рис. 3. Декомпозиция модельной сетки по 20 процессам

разделения расчетной области на подобласти, соответствующие разным процессам, минимизирующего количество обменов, в данной модели используется библиотека ParMETIS [10], интегрированная в INMOST. Сеточные данные, ассоциированные с элементами (вершина, ребро, треугольник), хранятся в упорядоченном формате, что позволяет использовать оптимальные алгоритмы поиска. На рис. 3 представлено разделение модельной сетки по 20 процессам.

Для параллельного решения систем линейных уравнений используется библиотека PETSc [11], также интегрированная в INMOST.

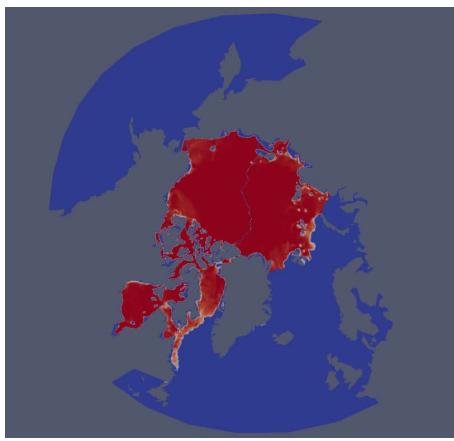


Рис. 4. Распределение сплоченности морского льда в 00:00 1 апреля 2020 г. по данным ТОРАЗ4, синтезированное на модельную сетку

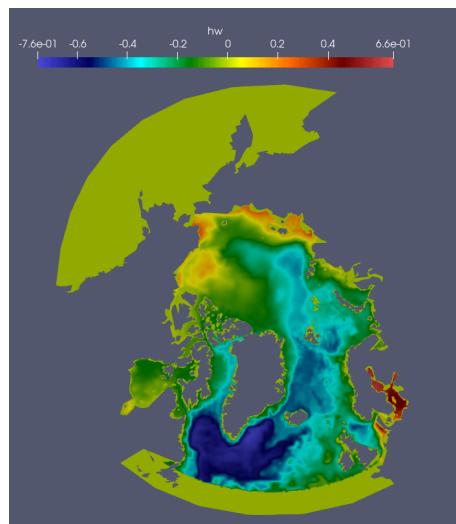


Рис. 5. Распределение уровня океана в 00:00 1 апреля 2020 г. по данным ТОРАЗ4, синтезированное на модельную сетку

Как уже отмечалось ранее, большинство геоданных распространяется в формате netCDF на прямоугольной сетке с несколькими уровнями. Для реализации динамического блока модели льда необходимо знать начальное распределение сплоченности и высоты льда, а также распределение уровня океана каждые несколько шагов интегрирования модели. Было принято решение насыщать модель данными из оперативной океанической европейской системы прогноза ТОРАЗ4 [12], которая входит в состав европейский программы оперативного прогноза состояния земной системы Copernicus [13]. Данный выбор обусловлен открытостью ресурса для некоммерческих исследований, регулярной обновляемостью и поддержкой. Данные ТОРАЗ4 поступают в оперативном режиме каждый час, что делает возможным валидацию разрабатываемой модели по этим данным в будущем. Функционал стандартной библиотеки netCDF [14] допускает параллельное считывание из netCDF-файла и запись в него. Данные ТОРАЗ-4 располагаются на прямоугольной сетке в стереографической проекции на касательную плоскость к Северному полюсу Земли. Каждый процессор считывает только необходимую ему часть прямоугольной сетки, что оптимизирует дальнейшие вычисления по времени и используемой памяти. Результаты параллельной билинейной интерполяции скалярных полей представлены на рис. 4 и рис. 5.

Помимо скалярных полей, для работы динамического ядра модели необходимо регулярное обновление внешнего океанического и атмосферного форсинга. Данные зональных компонент скоростей пограничного слоя атмосферы и океана представлены на рис. 6 и рис. 7 соответственно.

3. Оптимизированный метод численного интегрирования уравнения динамики морского льда с вязко-пластичной реологией. В динамических блоках современных глобальных моделей морского льда с вязко-пластичной реологией используются различные численные методы интегрирования. Наиболее распространены следующие два: mEVP [3], VP-Picard [15]. В данной работе проведено сравнение этих методов: качественное и по норме невязки в квадратной модельной области размера 1000 км с искусственным периодичным по времени и пространству внешним форсингом, который, несмотря на свою простоту, реализует вычислительно сложный режим медленного торочения льда. Также,

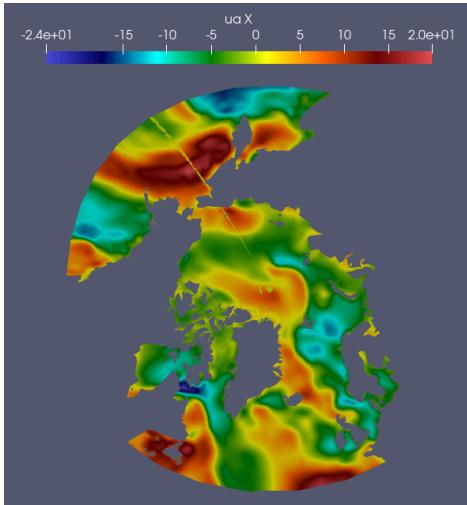


Рис. 6. Распределение зональной компоненты скорости ветра в 00:00 1 апреля 2020 г. по данным TOPAZ4, синтезированное на модельную сетку

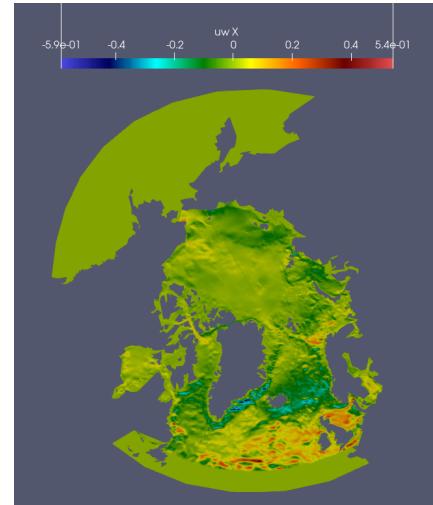


Рис. 7. Распределение зональной компоненты скорости воды в 00:00 1 апреля 2020 г. по данным TOPAZ4, синтезированное на модельную сетку

предложен альтернативный ускоренный метод mEVP-opt, основанный на классическом mEVP методе и идеи локального убывания квадрата нормы невязки линеаризованного оператора. Система двумерной динамики морского льда с вязко-пластичной реологией представляет из себя закон баланса импульса и два уравнения переноса — сплошности и массы льда [1]

$$\left\{ \begin{array}{l} m(\partial_t + f\mathbf{k} \times) \mathbf{u} = a\tau - C_d a \rho_0 (\mathbf{u} - \mathbf{u}_0) |\mathbf{u} - \mathbf{u}_0| + \mathbf{F} - mg\nabla H \\ \tau = C_a \rho_a |\mathbf{u}_a| \mathbf{u}_a \\ F_l = \frac{\partial \sigma_{kl}}{\partial x_k}, \quad l = 1, 2 \\ \sigma_{kl}(\mathbf{u}) = \frac{P}{2(\Delta + \Delta_{min})} \left[(\dot{\varepsilon}_d - \Delta) \delta_{kl} + \frac{1}{e^2} (2\dot{\varepsilon}_{kl} - \dot{\varepsilon}_d \delta_{kl}) \right] \\ \sigma_{1(2)} = \sigma_{11} \pm \sigma_{22}, \quad \varepsilon_{1(2)} = \sigma_{11} \pm \sigma_{22} \\ \dot{\varepsilon}_{kl} = \frac{1}{2} (\partial_k u_l + \partial_l u_k); \quad \dot{\varepsilon}_d = \dot{\varepsilon}_{kk} = \dot{\varepsilon}_{11} + \dot{\varepsilon}_{22} \\ \dot{\varepsilon}_s = ((\dot{\varepsilon}_{11} - \dot{\varepsilon}_{22})^2 + 4\dot{\varepsilon}_{12}^2)^{1/2}; \quad \Delta = (\dot{\varepsilon}_d^2 + \frac{1}{e^2} \dot{\varepsilon}_s^2)^{1/2} \\ P_0 = p^* h e^{-C(1-a)}; \quad P = \frac{P_0 \Delta}{\Delta + \Delta_{min}} \\ \partial_t a + \nabla \cdot (\mathbf{u} a) = 0, \quad a \leq 1 \\ \partial_t m + \nabla \cdot (\mathbf{u} m) = 0, \\ m = ah\rho, \end{array} \right. \quad (1)$$

где \mathbf{k} — единичный вертикальный вектор, f — параметр Корiolisa, m, a, h — масса, сплошность и высота льда, $\mathbf{u}, \mathbf{u}_0, \mathbf{u}_a$ — скорости льда, воды и воздуха, ρ, ρ_0, ρ_a — плотности льда, воды и воздуха, σ_{ij} — компоненты тензора напряжений, $\dot{\varepsilon}_{ij}$ — компоненты тензора скоростей деформации, $e = 2$ параметр эллиптичности, ξ, η — объемные и сдвиговые вязкости, P — давление, H — уровень океана.

Уравнения переноса решаются с помощью консервативной конечно-элементной схемы Тейлора-Галеркина с применением технологии коррекции потоков [16].

Суть mEVP-подхода [3] для решения уравнений динамики состоит в применении явной схемы Эйлера по времени отдельно по главным компонентам тензора напряжений и скоростям:

$$\begin{aligned} \alpha(\sigma_1^{p+1} - \sigma_1^p) &= \frac{P_0}{\Delta^p + \Delta_{min}}(\dot{\epsilon}_1^p - \Delta^p) - \sigma_1^p \\ \alpha(\sigma_2^{p+1} - \sigma_2^p) &= \frac{P_0}{\Delta^p + \Delta_{min}}\dot{\epsilon}_2^p - \sigma_2^p \\ \alpha(\sigma_{12}^{p+1} - \sigma_{12}^p) &= \frac{P_0}{\Delta^p + \Delta_{min}}\dot{\epsilon}_{12}^p - \sigma_{12}^p \\ \beta(\mathbf{u}^{p+1} - \mathbf{u}^p) &= -\mathbf{u}^{p+1} + \mathbf{u}^n - \Delta t \mathbf{f} \times \mathbf{u}^{p+1} + \\ &+ \frac{\Delta t}{m} [\mathbf{F}^{p+1} + a\tau + C_d a \rho_0 (\mathbf{u}_0^n - \mathbf{u}^{p+1}) |\mathbf{u}_0^p - \mathbf{u}^p| - mg \nabla H^n]. \end{aligned} \quad (2)$$

Здесь индекс p соответствует локальной итерации, а n — номер глобального шага по времени. Типичные значения параметров, которые используются в моделях: $\alpha = \beta = 500$ при количестве псевдоитераций равных $N_{\text{mEVP}} = 500$. Описанный процесс будем в дальнейшем называть mEVP-500. Главное преимущество такого подхода — отсутствие необходимости решать систему линейных уравнений на каждой итерации. Такое свойство получается за счет использования лампированной (диагонализованной) матрицы, путем суммирования по строке) массовой матрицы после применения метода Галеркина для пространственной аппроксимации.

Второй наиболее популярный подход для решения уравнения баланса импульса — это метод итераций Пикара [15]. Значения объемных и сдвиговых вязкостей берутся с предыдущей псевдоитерации и используются для формирования нового приближения компонент тензора напряжений. Затем эти компоненты подставляются непосредственно в уравнение баланса импульса, и вычисляются новые значения для скоростей. Таким образом организована неявная схема по времени, что позволяет избежать существенных ограничений на шаг интегрирования. Однако описанная схема требует решения системы линейных уравнений на каждой псевдоитерации, что существенно усложняет вычисления. Метод Пикара с 10-ю псевдоитерациями будем называть VP-10. На рис. 8 представлено сравнение относительной нормы невязки VP-10 и mEVP-500 методов. Видно, что решение, получаемое по второму методу, стремится к решению первого. Причем VP-10 дает более точный результат. Для сходимости требуется не менее 1400 итераций mEVP-500.

Дискретизованное по времени и пространству уравнение баланса импульса из (1) с помощью mEVP-метода (2) может быть записано в стандартном виде

$$\alpha(\mathbf{x}^{k+1} - \mathbf{x}^k) + A(\mathbf{x}^k)\mathbf{x}^k + \mathbf{f}(\mathbf{x}^k) = \mathbf{b}, \quad (3)$$

который представляет собой стационарный метод простой итерации с параметром α . Невязка уравнения записывается в виде

$$\mathbf{r}(\mathbf{x}) = A(\mathbf{x})\mathbf{x} + \mathbf{f}(\mathbf{x}) - \mathbf{b}.$$

Идея оптимизированного метода mEVP-opt состоит в том, чтобы организовать нестационарный метод простой итерации (3), где оптимальное значение параметра α_{opt} оценивается с помощью минимизации квадрата нормы невязки в линейном приближении [17]

$$\begin{aligned}
\|\mathbf{r}(\mathbf{x}^{k+1})\|^2 &= \|A(\mathbf{x}^{k+1})\mathbf{x}^{k+1} + \mathbf{f}(\mathbf{x}^{k+1}) - \mathbf{b}\|^2 = \\
&= \|A(\mathbf{x}^{k+1})[\mathbf{x}^k + \frac{1}{\alpha}(\mathbf{b} - A(\mathbf{x}^k)\mathbf{x}^k - \mathbf{f}(\mathbf{x}^k))] + \mathbf{f}(\mathbf{x}^{k+1}) - \mathbf{b}\|^2 \approx \\
&\approx \|\left[A(\mathbf{x}^k)\mathbf{x}^k + \mathbf{f}(\mathbf{x}^k) - \mathbf{b}\right] - \frac{1}{\alpha}A(\mathbf{x}^k)\left[A(\mathbf{x}^k)\mathbf{x}^k + \mathbf{f}(\mathbf{x}^k) - \mathbf{b}\right]\|^2 = \\
&= \|\mathbf{r}(\mathbf{x}^k) - \frac{1}{\alpha}A(\mathbf{x}^k)\mathbf{r}(\mathbf{x}^k)\|^2 = \\
&= \|\mathbf{r}(\mathbf{x}^k)\|^2 - \frac{2}{\alpha}(\mathbf{r}(\mathbf{x}^k), A(\mathbf{x}^k)\mathbf{r}(\mathbf{x}^k)) + \frac{1}{\alpha^2}\|A(\mathbf{x}^k)\mathbf{r}(\mathbf{x}^k)\|^2 \Rightarrow \\
&\Rightarrow \alpha_{\text{opt}} = \left[\frac{(\mathbf{r}(\mathbf{x}^k), A(\mathbf{x}^k)\mathbf{r}(\mathbf{x}^k))}{\|A(\mathbf{x}^k)\mathbf{r}(\mathbf{x}^k)\|^2} \right]^{-1}.
\end{aligned} \tag{4}$$

Выведенная оценка получается довольно грубой вследствие жесткого предположения о линейности функционала. Непосредственное использование оптимального параметра итерации на каждом шаге $\alpha = \alpha_{\text{opt}}$ (4) приводит к неустойчивости метода. Поэтому предлагается использовать следующую процедуру пересчета шага, начиная с $\alpha^0 = \alpha_{\text{def}}$ ($\alpha_{\text{def}} = 500$ в случае использования mEVP-500 метода)

$$\alpha^{k+1} = \alpha^k + \frac{(\alpha_{\text{opt}}^k - \alpha^k)}{C_\alpha \alpha_{\text{def}}}.$$

В вычислениях использовалось значение $C_\alpha = 2$. Также на очередной шаг α^{k+1} накладываются дополнительные ограничения

- 1) Убывание шага: $\alpha^{k+1} < \alpha^k$;
- 2) Относительное изменение шага не должно превышать некоторое наперед заданное число ε : $\frac{|\alpha^{k+1} - \alpha^k|}{\alpha_{\text{def}}} < \varepsilon$. В коде используется значение $\varepsilon = 0.05$;
- 3) Выполнение условия устойчивости mEVP-метода: $\alpha^2 > \frac{\pi^2 \cdot \max(P_0) \cdot \Delta t}{\Delta_{\min} \cdot \Delta x^2 \cdot \min(m)}$ [18].

При невыполнении любого из условий, перечисленных выше, очередной шаг присваивается равным предыдущему $\alpha^{k+1} = \alpha^k$.

Для проведения численного эксперимента была выбрана квадратная область размером $L = 1000$ км, что примерно соответствует реальным размерам Арктики. Начальные условия и внешний форсинг задаются аналогично [18]: $u_a = 5 + (\sin(\frac{2\pi t}{T}) - 3) \sin(\frac{2\pi x}{L}) \sin(\frac{\pi y}{L})$, $v_a = 5 + (\sin(\frac{2\pi t}{T}) - 3) \sin(\frac{2\pi y}{L}) \sin(\frac{\pi x}{L})$, $T = 4$ дня. Скорость воды задается как $u_0 = 0.1 \frac{2y-L}{L}$, $v_0 = -0.1 \frac{2x-L}{L}$. Уровень океана считается согласно геострофическому балансу. Высота льда равна 2 метра во всей области, а сплошность льда растет линейно от 0 до 1 в восточном направлении.

На рис. 9 показано преимущество mEVP-opt метода перед стандартным mEVP по невязке. Видно, что сходимость mEVP-opt достигается примерно за 800–1000 итераций.

Рис. 10 демонстрирует качественную картину лучшей сходимости mEVP-opt к решению VP-10, по сравнению со стандартным mEVP методом. Видно, что на верхней части рис. 10, b, область максимума скорости, полученная mEVP методом, не воспроизводится. Также можно заметить, что левая граница положительных скоростей завышена. Имея подобные проблемы, можно сделать вывод, что 800 итераций mEVP метода недостаточно для получения точного решения. Данных недостатков лишены распределения скорости, посчитанные mEVP-opt методом, представленные на рис. 10, c.

Заключение. В данной работе подробно описан процесс построения триангуляции Арктического региона со сгущением сетки в области потенциально высокой сплошности

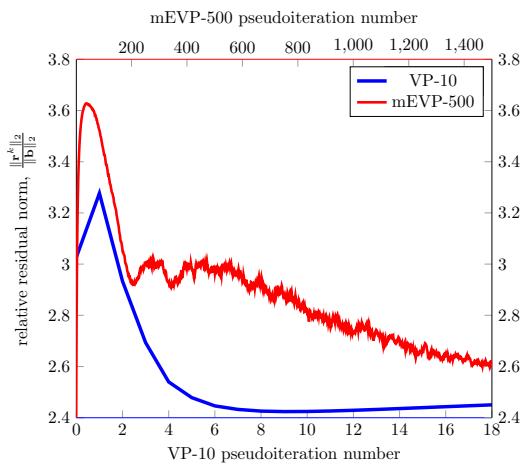


Рис. 8. Относительная норма невязки в зависимости от номера локальной итерации в момент времени $t=7$ часов. Синяя линия соответствует VP-10 методу, красная линия соответствует mEVP-500

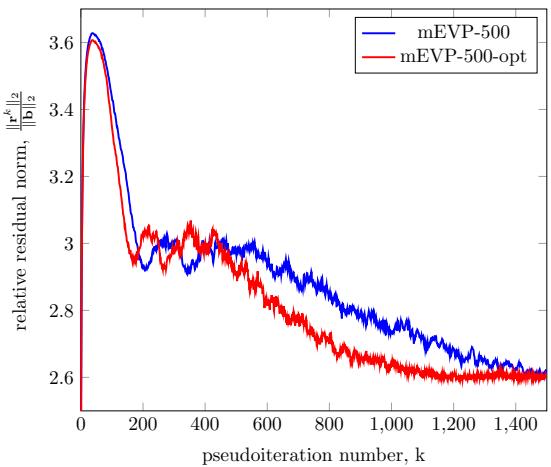


Рис. 9. Относительная норма невязки в зависимости от номера локальной итерации в момент времени $t=7$ часов. Синяя линия соответствует mEVP-500 методу, красная линия соответствует mEVP-500-opt

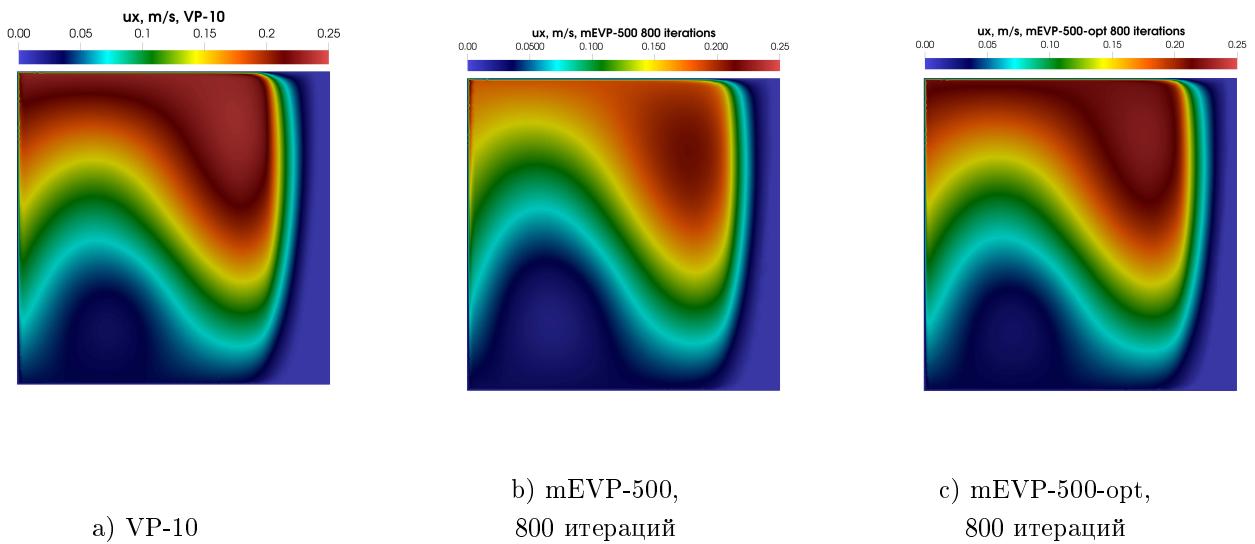


Рис. 10. Распределение зональной компоненты скорости льда спустя 7 часов моделирования

льда. Для построения триангуляции использовались пакет Ani2D и спутниковые данные сплошности льда за последние 10 лет. В основе динамического ядра разрабатываемой модели лежит пакет INMOST. Представлены результаты декомпозиции модельной сетки, интерполяции скалярных и векторных полей. Предложен оптимизированный метод mEVP-opt численного интегрирования уравнения баланса импульса морского льда с вязко-пластичной реологией. Качественно и количественно (по невязке) показано преимущество этого метода перед стандартным mEVP при незначительно больших вычислительных затратах.

Список литературы

1. Hibler W. D. A Dynamic Thermodynamic Sea Ice Model // Journal of Physical Oceanography. 1979. V. 9. N 4. P. 815–846.
2. Hunke E. C., Dukowicz J. K. An Elastic-Viscous-Plastic Model for Sea Ice Dynamics // Journal of Physical Oceanography. 1997. V. 27. N 9. P. 1849–1867.
3. Bouillon S., Fichefet T., Legat V., Madec G. The elastic-viscous-plastic method revisited // Ocean Modelling. 2013. V. 71. P. 2–12.
4. Kimmritz M., Danilov S., Losch M. The adaptive EVP method for solving the sea ice momentum equation // Ocean Modelling. 2016., V. 101.
5. Wessel P., Smith W. H. F. A Global Self-consistent, Hierarchical, High-resolution Shoreline Database // Journal of Geophysical Research. 2000. V. 101, P. 8741–8743.
6. Wessel P., Luis J.F., Uieda L., Scharroo R., Wobbe F., Smith W. H. F., Tian D. The Generic Mapping Tools version 6 // Geochemistry, Geophysics, Geosystems. 2019. V. 20. P. 5556–5564.
7. Danilov A. Unstructured tetrahedral mesh generation technology // Comp. Math. and Math. Phys. 2010. V. 50. N 1. P. 139–156.
8. Meier W. N., Fetterer F., Windnagel A. K. 2017. Near-Real-Time NOAA/NSIDC Climate Data Record of Passive Microwave Sea Ice Concentration. V. 1.
9. Danilov A., Terekhov K., Konshin I., Vassilevski Y. INMOST parallel platform: Framework for numerical modeling // Supercomputing Frontiers and Innovations. 2015. V. 2. P. 55–66.
10. Karypis G., Kumar V. A Fast and Highly Quality Multilevel Scheme for Partitioning Irregular Graphs // SIAM Journal on Scientific Computing. 1999. V. 20. N 1. P. 359–392.
11. [Электрон. Рес.]: <https://www.mcs.anl.gov/petsc/>
12. Sakov P., Counillon F., Bertino L., Lisjkter K. A., Oke P. R., Koralev A. TOPAZ4: an ocean-sea ice data assimilation system for the North Atlantic and Arctic // Ocean Sci. 2012. V. 8. P. 633–656.
13. [Электрон. Рес.]: <https://www.copernicus.eu/en>
14. [Электрон. Рес.]: <https://www.unidata.ucar.edu/software/netcdf/docs/index.html>
15. Hutchings J. K., Jasak H., Laxon S. W. A strength implicit correction scheme for the viscous-plastic sea ice model // Ocean Modell. 2004. V. 7. P. 111–133.
16. Kuzmin D., Turek S. Flux correction tools for finite elements // J. Comput. Phys. 2001. P. 525–558.
17. Ольшанский М. А. Лекции и упражнения по многосеточным методам // Физматлит. 2005. С. 24–25.
18. Danilov S., Wang Q., Timmermann R., Iakovlev N., Sidorenko D., Kimmritz M., Jung T., Schröter J. Finite-Element Sea Ice Model (FESIM), version 2 // Geosci. Model Dev. 2015. V. 8. P. 1747–1761.



Петров Сергей Сергеевич — аспирант Института Вычислительной Математики им. Г. И. Марчука (ИВМ РАН), e-mail: sergey.petrov@phystech.edu, телефон: +79266995520.

Начинающий специалист в области вычислительной математики, вычислительной геофизической гидродинамики, суперкомпьютерного моделирования, численных методов. Область научных интересов: океан, морской лед, метод

конечных элементов, многопроцессорные и многопоточные вычисления, вычисления на графических процессорах, разработка численных методов решения уравнений динамики морского льда с вязко-пластичной реологией, разработка численных методов решения уравнений термодинамики морского льда, методы усвоения данных, разработка прогностической системы состояния Арктического региона.

Sergey S. Petrov — Ph.D. student in Marchuk Institute of Numerical Mathematics of the Russian Academy of Sciences (INM RAS)

e-mail: sergey.petrov@phystech.edu, phone:
+79266995520

A beginner specialist in computational mathematics, computational geophysical fluid dynamics, supercomputer modeling, numerical methods. Research interests: ocean, sea ice, finite element method, multiprocessor and

multithreaded computing, GPU computing, development of numerical methods for solving equations of sea ice dynamics with viscous-plastic rheology, development of numerical methods for solving equations of thermodynamics of sea ice, methods of data assimilation, development of a predictive state system Arctic region.

Дата поступления — 03.02.2021



IMPLEMENTATION AND PERFORMANCE EVALUATION OF MULTIPLE PRECISION SPARSE MATRIX-VECTOR MULTIPLICATION ON CUDA USING THE RESIDUE NUMBER SYSTEM

K. Isupov, V. Knyazkov*, I. Babeshko, A. Krutikov

Vyatka State University,
610000, Kirov, Russia

*Penza State University,
440026, Penza, Russia

DOI: 10.24411/2073-0667-2021-10005

The solution of large systems of linear equations with sparse matrices is usually done using iterative methods that involve repeated computation of the sparse matrix-vector product (SpMV). The SpMV plays an important role in many scientific and engineering applications, and a lot of effort has been made to implement efficient SpMV kernels for highly parallel computing architectures such as graphics processing units (GPUs). However, round-off errors that occur when using finite precision floating-point arithmetic can cause poor convergence of iterative solvers. In this article, we present SpMV implementations for GPUs using multiple-precision floating-point arithmetic that are part of MPRES-BLAS, an open-source CUDA library of multiple-precision linear algebra operations (available at <https://github.com/kisupov/mpres-blas>).

We first give a brief description of the multiple-precision floating-point number representation supported in MPRES-BLAS. Unlike traditional approaches to expressing high-precision numbers, our representation is based on the residue number system (RNS). The RNS is an unweighted number system that is an alternative to the positional number representation. In the RNS, a number is represented by its residues modulo several pairwise relatively prime integers called the moduli. The RNS is interesting, in particular, because it provides efficient addition, subtraction and multiplication of large integers. the residues of a number are mutually independent, and the calculations with each residue are performed in the ring of integers modulo the corresponding modulus, so that carry chains between the residues are eliminated.

The SpMV performance depends on the format used to store the sparse matrix. In this article, we use the multiple-precision variants of CSR (Compressed Sparse Row) and ELLPACK sparse storage formats. The CSR format is widely used in iterative solvers, since it requires storing only nonzero matrix elements. In turn, the ELLPACK format is efficient when implemented on a GPU, provided the number of nonzero elements in each row of the matrix does not vary significantly.

Our multiple-precision SpMV algorithm consists of two steps. At the first step, the matrix and vector are multiplied element-by-element, and the computed array of component-wise products is stored in a global memory buffer. This step is highly parallelizable thanks to the use of RNS, namely, both all elements of the intermediate array and all digits of the significand (mantissa) for each multiple-precision element are computed in parallel. Each multiple-precision multiplication is performed by n GPU threads, where n is the size of the RNS moduli set. At the second step, the summation kernel is

This research was funded by the Russian Science Foundation grant number 20-71-00046.

launched, which perform a per-row reduction of the intermediate array to produce the output vector. In this step, each multiple-precision addition is performed by one thread.

In the experimental part of the paper, we evaluated the performance of our multiple-precision SpMV routines on the marine1, largebasis, ESOC, Goodwin_127 and nd6k matrices from SuiteSparse Matrix Collection (formerly known as the University of Florida Sparse Matrix Collection).

We compare the proposed SpMV with implementations based on existing multiple-precision general-purpose arithmetic libraries for CUDA-enabled GPUs, namely, CUMP and CAMPARY. We carry out the experiments on an NVIDIA RTX 2080 graphics card (46 streaming multiprocessors, 8 GB of GDDR6 memory, compute capability version 7.5). The results indicate that, with the same precision, the proposed SpMV routines in many cases provide significantly better performance than the CUMP and CAMPARY library implementations. One exception is the 106-bit precision case, where the CAMPARY library is much faster.

Comparison of the CSR and ELLPACK formats shows that ELLPACK outperforms CSR at a moderate level of precision (106–212 bits). With higher arithmetic precision (424–868 bits), the ELLPACK advantages become less apparent. Moreover, the memory requirements of ELLPACK can significantly exceed the requirements of CSR, especially in the context of higher precision. We note that matrices from real-world applications usually consist of single or double precision entries, and there is no need to convert them to higher precision. Considering this aspect, we plan to implement multiplication of a sparse double-precision matrix by a high-precision vector and use it to improve the convergence of iterative solvers.

Key words: Sparsematrices, SpMV, multiple-precisionarithmetic, residue number system, GPU programming.

References

1. Filippone S., Cardellini V., Barbieri D., Fanfarillo A. Sparse Matrix-Vector Multiplication on GPGPUs // ACM Transactions on Mathematical Software. 2017. Vol. 43, N 4. Article No. 30. DOI: 10.1145/3017994.
2. Bell N., Garland M. Efficient Sparse Matrix-Vector Multiplication on CUDA. Technical Report NVR-2008-004. NVIDIA Corporation. 2008. [Electron. Res.]: <https://www.nvidia.com/docs/IO/66889/nvr-2008-004.pdf> (date of access: 10.01.2021).
3. Greathouse J. L., Daga M. Efficient sparse matrix-vector multiplication on GPUs using the CSR storage format // Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC14, November 16–21, 2014, New Orleans, LA, USA. IEEE, 2014. P. 769–780. DOI: 10.1109/SC.2014.68.
4. Liu Y., Schmidt B. LightSpMV: Faster CSR-based sparse matrix-vector multiplication on CUDA-enabled GPUs // Proceedings of the 2015 IEEE 26th International Conference on Application-specific Systems, Architectures and Processors (ASAP), July 27–29, 2015, Toronto, ON, Canada. IEEE, 2015. P. 82–89. DOI: 10.1109/ASAP.2015.7245713.
5. Sedaghati N., Mu T., Pouchet L.-N., Parthasarathy S., Sadayappan P. Automatic selection of sparse matrix representation on GPUs // Proceedings of the 29th ACM on International Conference on Supercomputing, ICS'15, June 8–11, 2015 at Newport Beach, CA, USA. ACM, 2015. P. 99–108. DOI: 10.1145/2751205.2751244.
6. Cools S., Yetkin E. F., Agullo E., Giraud L., Vanroose W. Analyzing the Effect of Local Rounding Error Propagation on the Maximal Attainable Accuracy of the Pipelined Conjugate Gradient Method // SIAM Journal on Matrix Analysis and Applications. 2018. Vol. 39, N 1. P. 426–450. DOI: 10.1137/17M1117872.

7. Saito T., Ishiwata E., Hasegawa H. Analysis of the GCR method with mixed precision arithmetic using QuPAT // Journal of Computational Science. 2012. Vol. 3, N 3. P. 87–91. DOI: 10.1016/j.jocs.2011.05.001.
8. Shewchuk J. R. Adaptive Precision Floating-Point Arithmetic and Fast Robust Geometric Predicates // Discrete and Computational Geometry. 1997. Vol. 18, N 3. P. 305–363. DOI: 10.1007/PL00009321.
9. Masui K., Ogino M. Research on the Convergence of Iterative Method Using Mixed Precision Calculation Solving Complex Symmetric Linear Equation // IEEE Transactions on Magnetics. 2020. Vol. 56, N 1. Article no. 7503604. DOI: 10.1109/TMAG.2019.2951280.
10. Mukunoki D., Takahashi D. Using Quadruple Precision Arithmetic to Accelerate Krylov Subspace Methods on GPUs // Lecture Notes in Computer Science. 2014. Vol. 8384. P. 632–642. DOI: 10.1007/978-3-642-55224-3_59.
11. Hishinuma T., Hasegawa H., Tanaka T. SIMD Parallel Sparse Matrix-Vector and Transposed-Matrix-Vector Multiplication in DD Precision // Lecture Notes in Computer Science. 2017. Vol. 10150. P. 21–34. DOI: 10.1007/978-3-319-61982-8_4.
12. Kouya T. A highly efficient implementation of multiple precision sparse matrix-vector multiplication and its application to product-type Krylov subspace methods // International Journal of Numerical Methods and Applications. 2012. Vol. 7, N 2. P. 107–119.
13. Hirokawa Y., Itoh T., Tadano H., Ikuno S. Speedup and numerical evaluation of multiple-precision Krylov subspace method using GPU cluster for large-sparse linear system // Proceedings of the international conference for high performance computing, networking, storage and analysis, SC13, November 17–22, Denver, Colorado, USA. Colorado Convention Center, 2013.
14. Mohan Ananda P. A. Residue Number Systems. Theory and Applications. Cham: Birkhäuser. 2016.
15. Omondi A., Premkumar B. Residue Number Systems: Theory and Implementation. Imperial College Press, 2007.
16. Isupov K. Using Floating-Point Intervals for Non-Modular Computations in Residue Number System // IEEE Access. 2020. Vol. 8. P. 58603–58619. DOI: 10.1109/ACCESS.2020.2982365.
17. Isupov K., Knyazkov V., Kuvaev A. Design and Implementation of Multiple-Precision BLAS Level 1 Functions for Graphics Processing Units // Journal of Parallel and Distributed Computing. 2020. Vol. 140. P. 25–36. DOI: 10.1016/j.jpdc.2020.02.006.
18. Joldes M., Muller J. M., Popescu V. Implementation and Performance Evaluation of an Extended Precision Floating-Point Arithmetic Library for High-Accuracy Semidefinite Programming // Proceedings of the 24th IEEE Symposium on Computer Arithmetic, ARITH, July 24–26, 2017, London, UK. IEEE, 2017. P. 27–34. DOI: 10.1109/ARITH.2017.18.
19. Nakayama T., Takahashi D. Implementation of Multiple-Precision Floating-Point Arithmetic Library for GPU Computing // Proceedings of the 23rd IASTED International Conference on Parallel and Distributed Computing and Systems, PDCS 2011, December 14–16, 2011, Dallas, USA. ACTA Press, 2011. P. 343–349.



РЕАЛИЗАЦИЯ И ОЦЕНКА ПРОИЗВОДИТЕЛЬНОСТИ РАЗРЕЖЕННОГО МАТРИЧНО-ВЕКТОРНОГО УМНОЖЕНИЯ МНОГОКРАТНОЙ ТОЧНОСТИ НА CUDA С ИСПОЛЬЗОВАНИЕМ СИСТЕМЫ ОСТАТОЧНЫХ КЛАССОВ

К. С. Исупов, В. С. Князьков*, И. П. Бабешко, А. К. Крутиков

Вятский государственный университет,
610000, Киров, Россия

*Пензенский государственный университет,
440026, Пенза, Россия

УДК 004.222+004.272.25

DOI: 10.24411/2073-0667-2021-10005

Умножение разреженной матрицы на плотный вектор (SpMV) является основным и наиболее затратным элементом в итерационных методах решения разреженных линейных систем и задач на собственные значения. Эффективная реализация SpMV имеет решающее значение для многих научных и инженерных вычислений, причем важно обеспечить не только высокое быстродействие, но и достаточную точность SpMV, поскольку итерационные методы известны своей чувствительностью к ошибкам округления. В статье мы рассматриваем параллельные реализации SpMV для CUDA-совместимых графических процессоров видеокарт (GPU) с использованием арифметики многократной точности на основе системы остаточных классов (СОК). Основным преимуществом СОК перед позиционными системами счисления является отсутствие переносов между цифрами числа, что позволяет заменить многоразрядные операции группами покомпонентных операций с цифрами небольшой разрядности, которые выполняются без накладных расходов, связанных с обработкой информации о переносах между цифрами. Мы рассматриваем реализации SpMV, основанные на двух широко распространенных форматах хранения разреженной матрицы — CSR и ELLPACK. Экспериментальная оценка с матрицами из реальных приложений показывает, что во многих случаях представленные реализации выполняются быстрее, чем реализации на основе существующих библиотек многократной точности для GPU.

Ключевые слова: разреженные матрицы, SpMV, арифметика многократной точности, система остаточных классов, программирование для графических процессоров.

Введение. Операция разреженного матрично-векторного умножения (SpMV) занимает важное место в алгоритмах линейной алгебры, и многие исследования посвящены разработке форматов хранения разреженных матриц и реализаций SpMV, оптимизированных для параллельных вычислительных платформ, таких как центральные многоядерные процессоры (CPU) и графические процессоры видеокарт (GPU) [1–5]. При этом главным критерием выступает скорость вычислений. С другой стороны, точность SpMV также играет важную роль, поскольку итерационные методы, такие как метод минимальных

Исследование выполнено за счет гранта Российского научного фонда (проект № 20-71-00046).

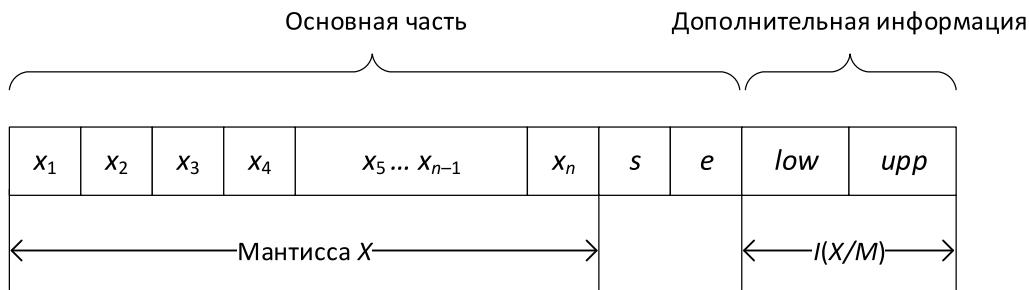


Рис. 1. Многоразрядный формат с плавающей точкой

невязок, метод сопряженных градиентов и метод бисопряженных градиентов, известны своей чувствительностью к ошибкам округления [6, 7].

Существуют различные способы повышения точности вычислений. Распространенным способом является использование формата double-double (DD) [8], в котором каждое число представляется в виде невычисленной суммы двух стандартных чисел с плавающей точкой двойной точности, x_h и x_l , где x_h — число, ближайшее к истинному значению, а x_l — разница между истинным значением и x_h . На основе DD-формата были предложены высокоточные версии SpMV, а также ряд других вычислительных ядер линейной алгебры и итерационных решателей [7, 9–11]. Другим подходом является использование арифметических библиотек многократной точности, таких как GMP/MPFR и CUMP [12, 13].

В статье мы рассматриваем реализации SpMV в форматах CSR (Compressed Sparse Row) и ELLPACK/ITPACK для CUDA-совместимых GPU с использованием арифметики многократной точности на основе непозиционной системы счисления с параллельной структурой — системы остаточных классов (СОК). Основным преимуществом СОК перед позиционными системами счисления является отсутствие переносов между цифрами числа, что позволяет заменить многоразрядные операции группами покомпонентных операций с цифрами небольшой разрядности, которые выполняются эффективно [14, 15]. Эксперименты с матрицами различной структуры показывают, что производительность предложенных реализаций SpMV во многих случаях выше, чем производительность реализаций на основе существующих CUDA-библиотек многократной точности CUMP и CAMPARY. Наши реализации SpMV являются частью библиотеки высокоточных подпрограмм линейной алгебры MPRES-BLAS и доступны на GitHub (<https://github.com/kisupov/mpres-blas>).

1. Представление многоразрядных чисел. Мы используем формат представления многоразрядных чисел (чисел многократной точности) с плавающей точкой, представленный на рис. 1. Число x представляется в виде объекта, состоящего из знака s , мантиссы X , целочисленного порядка (экспоненты) e , а также дополнительной информации о величине мантиссы $I(X/M)$.

Знак интерпретируется обычным образом: он равен нулю, если x — неотрицательное число, и единице, если x — отрицательное. Мантисса представлена в СОК цифрами (x_1, x_2, \dots, x_n) относительно заданного набора оснований (модулей) $\{m_1, m_2, \dots, m_n\}$ и интерпретируется как целое число в интервале от 0 до $M - 1$, где M — произведение всех модулей. Цифры мантиссы, которые в терминологии СОК называются остатками, являются такими целыми числами в дополнительном коде, что $x_i = X \bmod m_i$. Примеча-

тельной особенностью СОК является возможность эффективного выполнения операций сложения, вычитания и умножения: данные операции выполняются без возникновения переносов между соседними остатками, т. е. покомпонентно. Следовательно, в отличие от многоразрядной позиционной арифметики, не возникает накладных расходов, связанных с обработкой цепочек переносов, и, более того, все остатки могут вычисляться параллельно.

Преобразование из представленного числового формата в обычную позиционную форму основано на китайской теореме об остатках [15] и выполняется по формуле:

$$x = (-1)^s \times \left| \sum_{i=1}^n M_i |x_i w_i|_{m_i} \right|_M \times 2^e,$$

где $M_i = M/m_i$, а w_i — мультипликативная инверсия M_i по модулю m_i .

С другой стороны, имеются сложности, связанные с оценкой величины числа в СОК по остаткам, что требуется для определения знака, обнаружения переполнения диапазона представления мантиссы, сравнения, деления, округления и ряда прочих операций. Для преодоления этих сложностей в формат числа включена дополнительная информация — интервальная оценка мантиссы $I(X/M)$. Эта оценка представляет собой интервал, в пределах которого находится значение мантиссы X , масштабированное относительно произведения модулей M , т. е. X/M . Такой интервал задается двумя границами: *low* и *upp* на рис. 1. Границы являются числами с плавающей точкой стандартной разрядности, но с расширенным диапазоном экспоненты, что предотвращает потерю значимости в случае большой величины M (при $M < 2^{1000}$ для хранения *low* и *upp* достаточно стандартного формата двойной точности *binary64*).

Интервальная оценка вычисляется при преобразовании числа в многоразрядный формат, а при выполнении какой-либо арифметической операции с многоразрядными числами интервальная оценка результата вычисляется с использованием формул интервальной арифметики за время $O(1)$. Кроме этого, в любой момент интервальная оценка может быть заново вычислена по остаткам (x_1, x_2, \dots, x_n) с использованием лишь стандартных операций с плавающей точкой, т. е. без трудоемкого преобразования из СОК в двоичную систему счисления. Подробное изложение метода интервальных оценок для выполнения проблемных операций в СОК представлено в статье [16].

В [17] описаны алгоритмы выполнения основных арифметических операций многократной точности в представленном числовом формате, а также дана следующая граница погрешности округления: если $fl(x \circ y)$ — округленный результат операции $\circ \in \{+, -, \times\}$, то $fl(x \circ y) = (x \circ y)(1 + \delta)$, где $|\delta| < 4/\sqrt{M}$ и M — произведение всех модулей СОК. Таким образом, выбирая необходимый набор модулей СОК, возможно задавать произвольную точность используемой арифметики.

В библиотеке MPRES-BLAS для представления многоразрядных чисел и массивов чисел определены структуры данных, изображенные на рис. 2.

Структура `mp_float_t` представляет одно число. Для хранения в памяти GPU плотных многоразрядных векторов и матриц используется структура `mp_array_t`, в которой вектор из N элементов представляется в виде набора массивов, хранящих отдельные части многоразрядных чисел: все цифры многоразрядных мантисс хранятся в виде единого массива `digits` длины $n \times N$, где n — количество модулей СОК; знаки чисел, экспоненты и интервальные оценки мантисс хранятся в виде отдельных массивов. Дополнительно в структуру включены поля `len` (актуальная длина вектора) и `buf` (буфер с элементами векторного

```

typedef struct {
    int digits[n];
    int sign;
    int exp;
    er_float_t eval[2];
} mp_float_t;

typedef struct {
    int *digits;
    int *sign;
    int *exp;
    er_float_t *eval;
    int4 *buf;
    int *len;
} mp_array_t;

typedef struct {
    int *digits;
    int *sign;
    int *exp;
    er_float_t *eval;
} mp_collection_t;

```

Рис. 2. Структуры данных для представления многоразрядных чисел и массивов

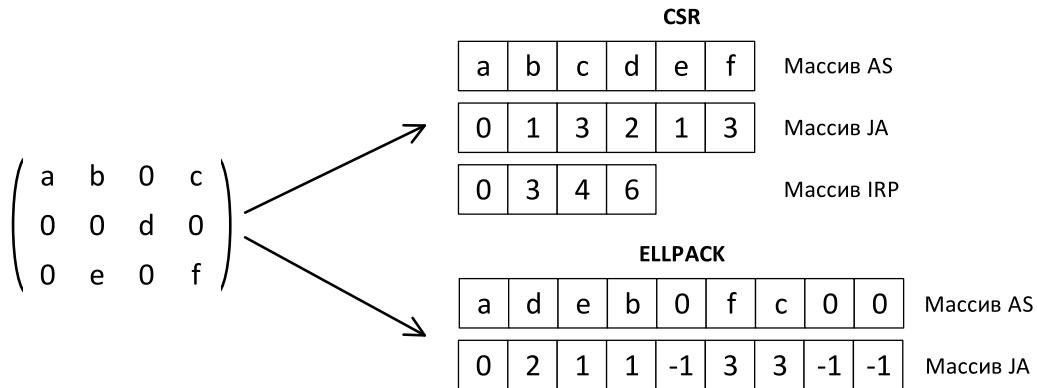


Рис. 3. Форматы CSR и ELLPACK. В качестве индексов нулевых элементов в ELLPACK могут использоваться контрольные значения, например –1

типа `int4` из стандартных заголовков CUDA C/C++, который используется для передачи вспомогательных переменных между вычислительными ядрами). Как показано в [17], использование структуры `mp_array_t` совместно с последовательной схемой адресации, при которой все n цифр одного многоразрядного числа располагаются последовательно в памяти GPU, обеспечивает при разрядно-параллельных вычислениях эффективный доступ параллельных потоков к памяти GPU. Для хранения разреженной многоразрядной матрицы служит структура `mp_collection_t`, аналогичная структуре `mp_array_t`, за исключением отсутствия полей `len` и `buf`, которые не используются в ядрах разреженной линейной алгебры.

2. Многоразрядные CSR и ELLPACK форматы. Производительность SpMV зависит от формата, используемого для хранения ненулевых коэффициентов разреженной матрицы. Основными являются координатный, диагональный, Compressed Sparse Row (CSR), ELLPACK/ITPACK и гибридный (HYB) форматы. На основе этих форматов предложены различные производные форматы и реализации SpMV, направленные на эффективное выполнение на GPU. Их обширный обзор представлен в [1]. В данной статье мы используем форматы CSR и ELLPACK, примеры которых приведены на рис. 3, где, подобно [1], AS — массив коэффициентов матрицы, JA — массив индексов столбцов и IRP — массив указателей начала строк.

Формат CSR получил наибольшее распространение в итерационных решателях. В этом формате размер массивов AS и JA равен количеству ненулевых элементов матрицы. Формат ELLPACK эффективен при реализации на GPU, при условии, что число ненулевых

CSR	ELLPACK
<pre> AS: { digits = [a.x1 a.x2 a.x3 a.x4 b.x1 b.x2 b.x3 b.x4 c.x1 c.x2 c.x3 c.x4 d.x1 d.x2 d.x3 d.x4 e.x1 e.x2 e.x3 e.x4 f.x1 f.x2 f.x3 f.x4] sign = [a.s b.s c.s d.s e.s f.s] exp = [a.e b.e c.e d.e e.e f.e] eval = [a.l b.l c.l d.l e.l f.l a.u b.u c.u d.u e.u f.u] } JA: [0 1 3 2 1 3] IRP: [0 3 4 6] </pre>	<pre> AS: { digits = [a.x1 a.x2 a.x3 a.x4 d.x1 d.x2 d.x3 d.x4 e.x1 e.x2 e.x3 e.x4 b.x1 b.x2 b.x3 b.x4 0 0 0 0 f.x1 f.x2 f.x3 f.x4 c.x1 c.x2 c.x3 c.x4 0 0 0 0 0 0 0 0] sign = [a.s d.s e.s b.s 0 f.s c.s 0 0] exp = [a.e d.e e.e b.e 0 f.e c.e 0 0] eval = [a.l d.l e.l b.l 0 f.l c.l 0 0 a.u d.u e.u b.u 0 f.u c.u 0 0] } JA: [0 2 1 1 -1 3 3 -1 -1] </pre>

Рис. 4. Многоразрядные форматы CSR и ELLPACK

элементов в каждой строке матрицы варьируется незначительно [2]. Размер массивов AS и JA в ELLPACK равен $M \times MAXNZR$, где M — число строк и $MAXNZR$ — максимальное число ненулевых элементов в строке матрицы.

На рис. 4 приведен пример представления разреженной матрицы в многоразрядных CSR и ELLPACK форматах. В примере количество модулей СОК $n = 4$, т. е. мантисса каждого элемента $\circ \in \{a,b,c,d,e,f\}$ состоит из четырех остатков: $X = (x_1, x_2, x_3, x_4)$. Символ «.» используется для доступа к отдельным полям многоразрядного числа, а l и u обозначают соответственно нижнюю и верхнюю границы интервальной оценки мантиссы (*low* и *upp* на рис.).

3. GPU реализация SpMV многократной точности. Входными данными для наших реализаций SpMV ($y \leftarrow Ax$) являются разреженная матрица A , состоящая из M строк и N столбцов, содержащая NNZ ненулевых элементов, а также плотный вектор x размера N , хранящийся в виде экземпляра структуры `mp_array_t`. Ненулевые элементы матрицы (массив AS) хранятся в виде экземпляра структуры `mp_collection_t`. Дополнительно требуется выделить в глобальной памяти GPU массив BUF (`mp_collection_t`), размер которого равен NNZ для CSR формата и $M \times MAXNZR$ для ELLPACK формата. Результатом SpMV является плотный многоразрядный вектор y размера M (структуре `mp_array_t`).

Выполнение операции состоит из двух шагов:

- 1) Вычисление массива покомпонентных произведений матрицы A и вектора x с сохранением результатов в промежуточный массив BUF .
- 2) Формирование вектора y путем суммирования массива BUF по строкам.

Использование СОК позволяет реализовать полностью параллельную схему выполнения первого шага SpMV, в соответствии с которой одновременно вычисляются не только элементы массива BUF , но также все цифры многоразрядной мантиссы каждого элемента. Для выполнения каждого умножения многократной точности используется n потоков GPU, где n — размер набора модулей СОК. Для эффективной реализации данной схемы

мы операция многоразрядного умножения разбивается на три этапа, каждый из которых выполняется отдельным вычислительным ядром (глобальной функцией GPU), как предложено в [17]: (1) ядро для вычисления знаков, экспонент и интервальных оценок мантисс, (2) ядро для полностью параллельного вычисления цифр многоразрядных мантисс и (3) ядро для округления.

Таким образом, выполнение SpMV многократной точности состоит из запуска четырех CUDA ядер: три ядра для умножения и одно для построчной редукции промежуточного массива.

В алгоритмах 1–3 представлен псевдокод первых двух ядер умножения и ядра редукции в CSR формате. В алгоритмах 4–6 представлен псевдокод соответствующих ядер для ELLPACK формата, в которых используется проверка контрольного значения индекса столбца для исключения ненужных обращений к памяти. Функции *RoundDown* и *RoundUp* обозначают вычисления с плавающей точкой стандартной точности (double) с расширенным диапазоном экспоненты, выполняемые с округлением „вниз“ и „вверх“ соответственно.

Алгоритм 1. Вычисление знаков, экспонент и интервальных оценок в CSR формате

```

1: i = blockDim.x * blockIdx.x + threadIdx.x
2: while i < NNZ do
3:   id = JA(i)
4:   BUF.sign(i) = AS.sign(i) xor x.sign(id)
5:   BUF.exp(i) = AS.exp(i) + x.exp(id)
6:   BUF.eval(i) = RoundDown(AS.eval(i)) * x.eval(id) / ONE.upp
7:   BUF.eval(NNZ + i) = RoundUp(AS.eval(NNZ + i)) * x.eval(N + id) / ONE.low
8:   i += blockDim.x * blockDim.x
9: end while
```

Алгоритм 2. Вычисление цифр многоразрядных мантисс в CSR формате

```

1: tid = threadIdx.x mod n                                ▷ n — размер набора модулей
2: m = MODULI(tid)
3: i = blockIdx.x * blockDim.x + threadIdx.x
4: j = (blockIdx.x * blockDim.x + threadIdx.x) / n
5: while i < NNZ * n do
6:   BUF.digits(i) = (AS.digits(i) * x.digits(JA(j) * n + tid)) mod m
7:   i += blockDim.x * blockDim.x
8:   j += blockDim.x * blockDim.x / n
9: end while
```

Алгоритм 3. Редукция промежуточного массива BUF в CSR формате

```

1: row = threadIdx.x + blockIdx.x * blockDim.x
2: if row < M then
3:   sum(threadIdx.x) = 0
4:   for i = IRP(row):IRP(row + 1) do
5:     sum(threadIdx.x) += BUF(i)                                ▷ Многоразрядное сложение
6:   end for
7:   y(row) = sum(threadIdx.x)
8: end if
```

Алгоритм округления и его CUDA-реализация ранее описаны в работе [17] и поэтому в данной статье не рассматриваются.

Алгоритм 4. Вычисление знаков, экспонент и интервальных оценок в ELLPACK формате

```

1: i = blockDim.x * blockIdx.x + threadIdx.x
2: len = M * MAXNZR
3: while i < len do
4:   id = JA(i)
5:   if id ≥ 0 then
6:     BUF.sign(i) = AS.sign(i) xor x.sign(id)
7:     BUF.exp(i) = AS.exp(i) + x.exp(id)
8:     BUF.eval(i) = RoundDown(AS.eval(i) * x.eval(id) / ONE.upp)
9:     BUF.eval(len + i) = RoundUp(AS.eval(len + i) * x.eval(N + id) / ONE.low)
10:    end if
11:    i += blockDim.x * blockDim.x
12: end while

```

Алгоритм 5. Вычисление цифр многоразрядных мантисс в ELLPACK формате

```

1: tid = threadIdx.x mod n                                ▷ n — размер набора модулей
2: m = MODULI(tid)
3: i = blockIdx.x * blockDim.x + threadIdx.x
4: j = (blockIdx.x * blockDim.x + threadIdx.x) / n
5: while i < M * MAXNZR * n do
6:   id = JA(j)
7:   if id ≥ 0 then
8:     BUF.digits(i) = (AS.digits(i) * x.digits(id * n + tid)) mod m
9:   end if
10:  i += blockDim.x * blockDim.x
11:  j += blockDim.x * blockDim.x / n
12: end while

```

Алгоритм 6. Редукция промежуточного массива BUF в ELLPACK формате

```

1: row = threadIdx.x + blockIdx.x * blockDim.x
2: if row < M then
3:   sum(threadIdx.x) = 0
4:   for i = 0:MAXNZR do
5:     j = i * M + row
6:     id = JA(j)
7:     if id ≥ 0 then
8:       sum(threadIdx.x) += BUF(j)                         ▷ Многоразрядное сложение
9:     end if
10:   end for
11:   y(row) = sum(threadIdx.x)
12: end if

```

4. Оценка производительности и затрат памяти. Производительность рассмотренных реализаций SpMV оценивалась на матрицах различной структуры из коллекции Университета Флориды (<https://sparse.tamu.edu>). Описание матриц представлено в табл. 1. Символы M , N , NNZ и $MAXNZR$ обозначают, соответственно, число строк, число столбцов, общее число ненулевых элементов матрицы и максимальное число ненулевых элементов в строке.

Эксперименты выполнялись на вычислительной системе с видеокартой NVIDIA RTX 2080 GPU (46 потоковых мультипроцессоров, 8 Гб памяти GDDR6, версия аппаратной совместимости 7.5), центральным процессором Intel Core i5 7500 и 16 Гб оперативной памяти DDR4, работающей под управлением операционной системы Ubuntu 20.04.1 LTS. Исполь-

Таблица 1
Матрицы для экспериментов

Матрица	<i>M</i>	<i>N</i>	<i>NNZ</i>	<i>MAXNZR</i>
marine1	400 320	400 320	6 226 538	18
largebasis	440 020	440 020	5 240 084	14
ESOC	327 062	37 830	6 019 939	19
Goodwin_127	178 437	178 437	5 778 545	62
nd6k	18 000	18 000	6 897 316	514

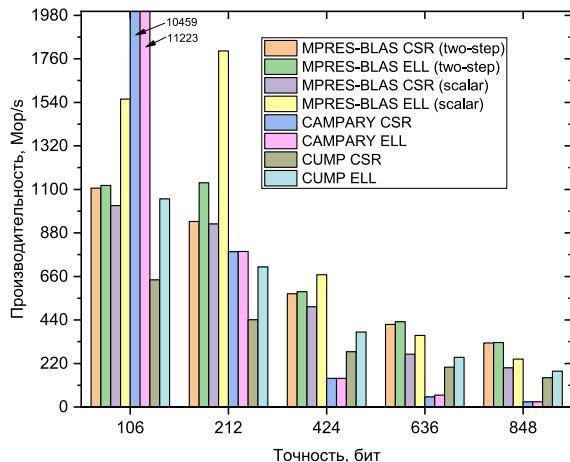
зовались CUDA Toolkit версии 11.1.105 и драйвер NVIDIA версии 455.32.00. Исходный код компилировался с опцией *-O3*.

Оценивалось восемь реализаций SpMV многократной точности, построенных на базе высокоточных CUDA-библиотек MPRES-BLAS, CAMPARY [18] и CUMP [19]:

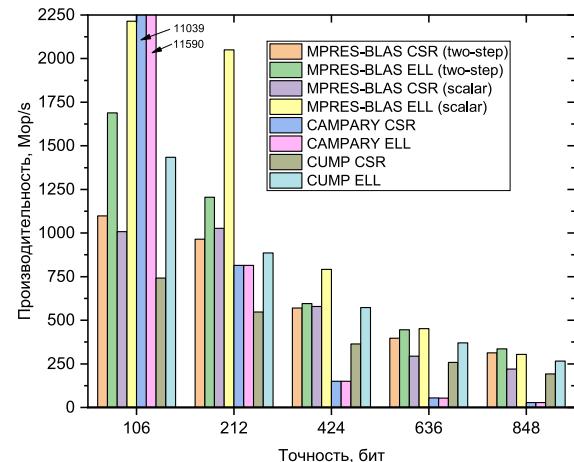
1) *MPRES-BLAS CSR (two-step)* — рассмотренная в предыдущем разделе двухшаговая реализация из библиотеки MPRES-BLAS с использованием формата CSR; 2) *MPRES-BLAS ELL (two-step)* — рассмотренная в предыдущем разделе двухшаговая реализация из библиотеки MPRES-BLAS с использованием формата ELLPACK; 3) *MPRES-BLAS CSR (scalar)* — реализация из библиотеки MPRES-BLAS, полученная из обычного CSR ядра SpMV заменой стандартных операций с плавающей точкой многоразрядными операциями; каждая строка матрицы обрабатывается одним GPU потоком; (см. [2], рис. 20); матрица и векторы хранятся в виде массивов многоразрядных чисел типа `mp_float_t`; 4) *MPRES-BLAS ELL (scalar)* — реализация из библиотеки MPRES-BLAS, полученная из обычного ELLPACK ядра SpMV заменой стандартных операций с плавающей точкой многоразрядными операциями; каждая строка матрицы обрабатывается одним GPU потоком; (см. [2], рис. 17); матрица и векторы хранятся в виде массивов многоразрядных чисел типа `mp_float_t`; 5) *CAMPARY CSR* — реализация CSR ядра с использованием библиотеки CAMPARY; 6) *CAMPARY ELL* — реализация ELLPACK ядра с использованием CAMPARY; 7) *CUMP CSR* — реализация CSR ядра с использованием библиотеки CUMP; 8) *CUMP ELL* — реализация ELLPACK ядра с использованием библиотеки CUMP.

Производительность измерялась при арифметической точности от 106 до 848 бит. Для достижения заданной точности в MPRES-BLAS использовалось от 8 до 64 модулей СОК. Мерой производительности *P* является Мор/с — миллион многоразрядных операций с плавающей точкой в секунду: $P = 2NNZ/(1000T)$, где *T* — измеренное время выполнения операции в мс. Заметим, что каждая многоразрядная операция состоит из ряда стандартных операций. Измерялось только время вычислений, т. е. не учитывались время преобразования данных в многоразрядные форматы и время копирования в память GPU. Полученные результаты представлены на рис. 4.

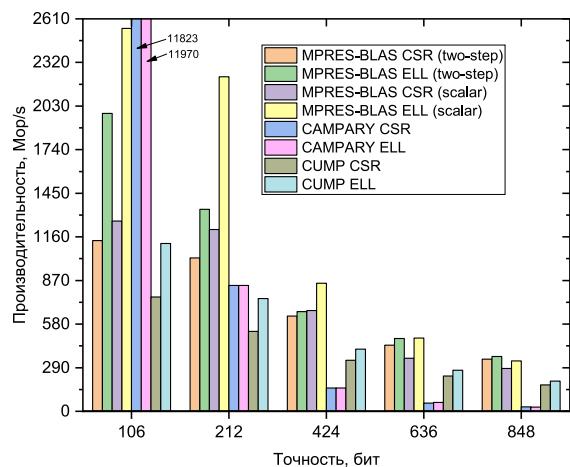
Эффективность многоразрядных реализаций SpMV зависит как от точности, так и от структуры матрицы. Лучшая производительность достигается на матрицах, состоящих из большого числа коротких строк (marine1, largebasis и ESOC), так как такая структура обеспечивает лучшую масштабируемость вычислительных ядер за счет мелкозернистого параллелизма: все строки могут обрабатываться параллельно, и объем вычислений для каждого потока относительно небольшой. В свою очередь, матрицы Goodwin_127 и nd6k состоят из меньшего числа строк, но каждая строка содержит большее число элементов,



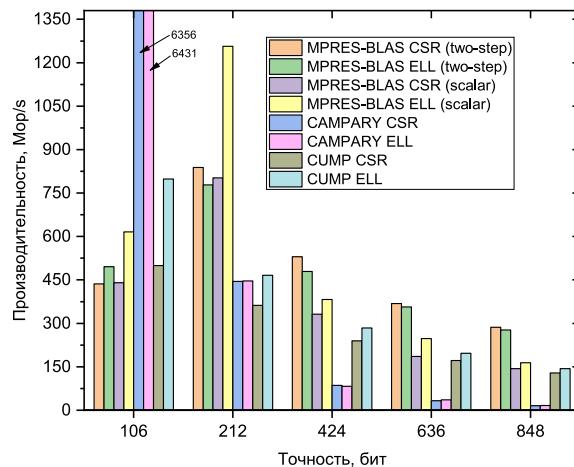
(a) marinel



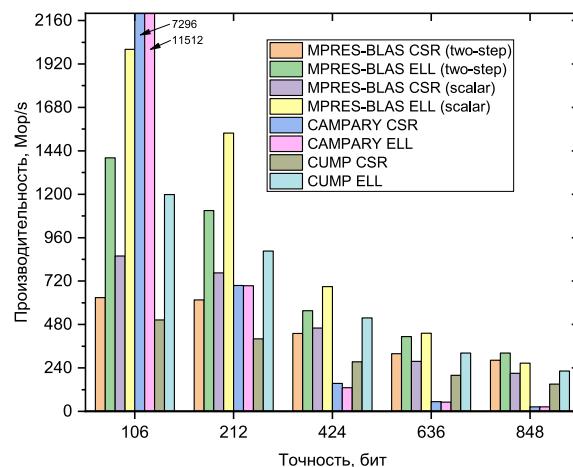
(b) largebasis



(c) ESOC



(d) Goodwin_127



(e) nd6k

Рис. 4. Производительность SpMV многократной точности на NVIDIA RTX 2080

Таблица 2

Размер (МБ) многоразрядной матрицы в форматах CSR и ELLPACK

Точность	marine1		largebasis		ESOC		Goodwin_127		nd6k	
	CSR	ELL	CSR	ELL	CSR	ELL	CSR	ELL	CSR	ELL
106	427	494	381	422	413	426	396	759	473	635
212	617	714	551	610	597	616	573	1097	684	917
424	997	1154	890	986	964	995	925	1772	1105	1482
636	1377	1594	1230	1362	1331	1374	1278	2447	1526	2047
848	1757	2034	1569	1738	1699	1754	1631	3122	1947	2611

что приводит к меньшей производительности, так как увеличивается объем вычислений, выполняемый каждым потоком GPU.

При 106-битной точности самой быстрой оказалась библиотека CAMPARY, реализующая оптимизированные алгоритмы double-double арифметики (для этой точности в экспериментах использовался формат 2-double CAMPARY). Однако при увеличении точности производительность CAMPARY существенно снижается, и при точности от 212 до 848 бит лучшие результаты показывают реализации на основе библиотеки MPRES-BLAS.

При точности от 106 до 424 бит рассмотренные в предыдущем разделе двухшаговые реализации имеют производительность, сравнимую со скалярными SpMV ядрами на основе `mp_float_t`, либо проигрывают им. Однако при более высокой точности производительность двухшаговых реализаций, как правило, выше, чем производительность скалярных ядер. Наибольшее ускорение ($2\times$) достигнуто при точности 848 бит на матрице Goodwin_127 с использованием CSR формата. Также следует отметить, что на матрице Goodwin_127 производительность 106-битных MPRES-BLAS реализаций ниже, чем производительность 212-битных MPRES-BLAS реализаций. Это связано с большим количеством округлений, возникающих в процессе вычислений. Оптимизация округления является важным направлением работ по развитию библиотеки MPRES-BLAS.

В табл. 2 приведен размер массива ненулевых элементов многоразрядной матрицы для реализаций SpMV, рассмотренных в разделе 3. Стоит заметить, что такой же размер имеет буфер, необходимый для хранения промежуточного массива произведений.

Заключение. В работе рассмотрены реализации разреженного матрично-векторного умножения (SpMV) для графических процессоров видеокарт в арифметике многократной точности с использованием системы остаточных классов (СОК), являющиеся частью библиотеки высокоточных подпрограмм линейной алгебры MPRES-BLAS. Непозиционная структура СОК исключает переносы между соседними цифрами многоразрядных мантисс и позволяет вычислять все цифры параллельно. Экспериментальные результаты свидетельствуют о том, что при одинаковой точности разработанные версии SpMV во многих случаях обеспечивают лучшее быстродействие по сравнению с реализациями, основанными на позиционных библиотеках CUMP и CAMPARY. Сравнение форматов CSR и ELLPACK показывает, что ELLPACK превосходит CSR при небольшой точности (106–212 бит). При более высокой точности преимущества ELLPACK становятся менее очевидными, и, более того, затраты памяти ELLPACK могут существенно превышать затраты CSR, в особенности когда речь идет о многоразрядных матрицах.

В дальнейшем запланирована реализация векторных CSR ядер многократной точности, в которых каждая строка матрицы обрабатывается группой потоков GPU. Также

стоит отметить, что матрицы, возникающие в реальных приложениях, как правило, представлены в формате двойной точности (double) и выполнять их преобразование в многоразрядный формат не обязательно. Поэтому в дальнейшем, в рамках развития библиотеки MPRES-BLAS планируются реализация и оценка эффективности операций умножения разреженной матрицы двойной точности на вектор многократной точности, что позволит существенно снизить расходы памяти GPU.

Список литературы

1. Filippone S., Cardellini V., Barbieri D., Fanfarillo A. Sparse Matrix-Vector Multiplication on GPGPUs // ACM Transactions on Mathematical Software. 2017. Vol. 43, N 4. Article No. 30. DOI: 10.1145/3017994.
2. Bell N., Garland M. Efficient Sparse Matrix-Vector Multiplication on CUDA. Technical Report NVR-2008-004. NVIDIA Corporation. 2008. [Electron. Res.]: <https://www.nvidia.com/docs/IO/66889/nvr-2008-004.pdf> (date of access: 10.01.2021).
3. Greathouse J. L., Daga M. Efficient sparse matrix-vector multiplication on GPUs using the CSR storage format // Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC14, November 16–21, 2014, New Orleans, LA, USA. IEEE, 2014. P. 769–780. DOI: 10.1109/SC.2014.68.
4. Liu Y., Schmidt B. LightSpMV: Faster CSR-based sparse matrix-vector multiplication on CUDA-enabled GPUs // Proceedings of the 2015 IEEE 26th International Conference on Application-specific Systems, Architectures and Processors (ASAP), July 27–29, 2015, Toronto, ON, Canada. IEEE, 2015. P. 82–89. DOI: 10.1109/ASAP.2015.7245713.
5. Sedaghati N., Mu T., Pouchet L.-N., Parthasarathy S., Sadayappan P. Automatic selection of sparse matrix representation on GPUs // Proceedings of the 29th ACM on International Conference on Supercomputing, ICS'15, June 8–11, 2015 at Newport Beach, CA, USA. ACM, 2015. P. 99–108. DOI: 10.1145/2751205.2751244.
6. Cools S., Yetkin E.F., Agullo E., Giraud L., Vanroose W. Analyzing the Effect of Local Rounding Error Propagation on the Maximal Attainable Accuracy of the Pipelined Conjugate Gradient Method // SIAM Journal on Matrix Analysis and Applications. 2018. Vol. 39, N 1. P. 426–450. DOI: 10.1137/17M1117872.
7. Saito T., Ishiwata E., Hasegawa H. Analysis of the GCR method with mixed precision arithmetic using QuPAT // Journal of Computational Science. 2012. Vol. 3, N 3. P. 87–91. DOI: 10.1016/j.jocs.2011.05.001.
8. Shewchuk J. R. Adaptive Precision Floating-Point Arithmetic and Fast Robust Geometric Predicates // Discrete and Computational Geometry. 1997. Vol. 18, N 3. P. 305–363. DOI: 10.1007/PL00009321.
9. Masui K. Ogino M. Research on the Convergence of Iterative Method Using Mixed Precision Calculation Solving Complex Symmetric Linear Equation // IEEE Transactions on Magnetics. 2020. Vol. 56, N 1. Article no. 7503604. DOI: 10.1109/TMAG.2019.2951280.
10. Mukunoki D., Takahashi D. Using Quadruple Precision Arithmetic to Accelerate Krylov Subspace Methods on GPUs // Lecture Notes in Computer Science. 2014. Vol. 8384. P. 632–642. DOI: 10.1007/978-3-642-55224-3_59.
11. Hishinuma T., Hasegawa H., Tanaka T. SIMD Parallel Sparse Matrix-Vector and Transposed-Matrix-Vector Multiplication in DD Precision // Lecture Notes in Computer Science. 2017. Vol. 10150. P. 21–34. DOI: 10.1007/978-3-319-61982-8_4.
12. Kouya T. A highly efficient implementation of multiple precision sparse matrix-vector multiplication and its application to product-type Krylov subspace methods // International Journal of Numerical Methods and Applications. 2012. Vol. 7, N 2. P. 107–119.

13. Hirokawa Y., Itoh T., Tadano H., Ikuno S. Speedup and numerical evaluation of multiple-precision Krylov subspace method using GPU cluster for large-sparse linear system // Proceedings of the international conference for high performance computing, networking, storage and analysis, SC13, November 17–22, Denver, Colorado, USA. Colorado Convention Center, 2013.
14. Mohan Ananda P. A. Residue Number Systems. Theory and Applications. Cham: Birkhäuser. 2016.
15. Omondi A., Premkumar B. Residue Number Systems: Theory and Implementation. Imperial College Press, 2007.
16. Isupov K. Using Floating-Point Intervals for Non-Modular Computations in Residue Number System // IEEE Access. 2020. Vol. 8. P. 58603–58619. DOI: 10.1109/ACCESS.2020.2982365.
17. Isupov K., Knyazkov V., Kuvayev A. Design and Implementation of Multiple-Precision BLAS Level 1 Functions for Graphics Processing Units // Journal of Parallel and Distributed Computing. 2020. Vol. 140. P. 25–36. DOI: 10.1016/j.jpdc.2020.02.006.
18. Joldes M., Muller J. M., Popescu V. Implementation and Performance Evaluation of an Extended Precision Floating-Point Arithmetic Library for High-Accuracy Semidefinite Programming // Proceedings of the 24th IEEE Symposium on Computer Arithmetic, ARITH, July 24–26, 2017, London, UK. IEEE, 2017. P. 27–34. DOI: 10.1109/ARITH.2017.18.
19. Nakayama T., Takahashi D. Implementation of Multiple-Precision Floating-Point Arithmetic Library for GPU Computing // Proceedings of the 23rd IASTED International Conference on Parallel and Distributed Computing and Systems, PDCS 2011, December 14–16, 2011, Dallas, USA. ACTA Press, 2011. P. 343–349.



Исупов Константин Сергеевич — канд. технич. наук, доцент кафедры электронных вычислительных машин Вятского государственного университета, e-mail: ks_isupov@vyatsu.ru. Область профессиональных интересов: вычисления многократной точности, система остаточных классов, компьютерная арифметика, параллельные алгоритмы, программирование для графических процессоров, CUDA. Является автором и соавтором более 30 научных работ в указанных областях.

Konstantin Isupov, PhD in Computer Science. Assistant professor at the Department of Electronic Computing Machines, Vyatka State University, e-mail: ks_isupov@vyatsu.ru. His research interests include high-precision computation, residue number system, computer arithmetic, parallel algorithms, GPU programming, CUDA. He has authored or coauthored more than 30 papers in these areas.

Князьков Владимир Сергеевич — д-р технич. наук, профессор, главный научный сотрудник научно-исследовательского института

фундаментальных и прикладных исследований Пензенского государственного университета, e-mail: knyazkov@pnzgu.ru. Является автором более 50 свидетельств на изобретения и патентов, опубликовал более 100 научных работ в области параллельных архитектур, высокопроизводительных вычислений и реконфигурируемых вычислительных систем.



Vladimir Knyazkov, Dr. Sci. in Computer Sciences and Engineering. Principal research scientist at the Research Institute of Fundamental and Applied Studies, Penza State University, e-mail: knyazkov@pnzgu.ru. He is the author of 50 invention certificates and patents, and published over 100 papers in the fields of parallel architectures, high-performance computing and reconfigurable computing systems.

Бабешко Иван Павлович — аспирант кафедры электронных вычислительных машин Вятского государственного университета, e-mail: stud120279@vyatsu.ru. Область научных интересов: высокоточные вычисления, арифме-

тика в остаточных классах, программирование для графических процессоров.



Ivan Babeshko is a postgraduate student at the Department of Electronic Computing Machines, Vyatka State University, e-mail: stud120279@vyatsu.ru. Educational degrees: Master's degree in Informatics and Computer Engineering, 2019, Vyatka State University. His current research interests are multiple-precision arithmetic, residue number system, GPU programming.

Крутиков Александр Константинович — аспирант кафедры электронных вычислительных машин Вятского государственного

университета, e-mail: usr09603@vyatsu.ru. Область научных интересов: прогнозирование методами и средствами искусственного интеллекта, искусственные нейронные сети, высокопроизводительные вычисления.



Alexander Krutikov is a postgraduate student at the Department of Electronic Computing Machines, Vyatka State University, e-mail: usr09603@vyatsu.ru. Educational degrees: Master's degree in Informatics and Computer Engineering, 2018, Vyatka State University. His current research interests are artificial neural networks and high-performance computing.

Дата поступления — 03.02.2021

SHALLOW WATER MODEL USING A HYBRID MPI/OPENMP PARALLEL PROGRAMMING

A. V. Chaplygin, A. V. Gusev

Marchuk Institute of Numerical Mathematics, Russian Academy of Sciences,
119333, Moscow, Russia

DOI: 10.24411/2073-0667-2021-10006

Hybrid models that combine MPI message passing technologies for distributed memory architectures and OpenMP multithreading for shared memory architectures are becoming increasingly popular. Modern high-performance computing systems are multiprocessor systems with shared memory (computing nodes), united in a single communication network. Creating models that effectively use the resources of such computing systems is an urgent task today. The portability of models to multiprocessor systems is largely determined by the correct choice of its software architecture, which would also provide flexibility and ease of maintaining the program code.

The paper considers the system of shallow water equations in the form presented in the ocean general circulation sigma model INMOM (Institute of Numerical Mathematics Ocean Model). The INMOM model is being developed at the INM RAS and is used as a ocean component of the INMCM (Institute of Numerical Mathematics Climate Model) climate model, created at the INM RAS and participating in the IPCC (Intergovernmental Panel on Climate Change) program for climate change forecasting. The model is completely written in the Fortran 90/95 programming language. In previous works, a shallow water model has been formulated that can be used both as a program block of the sigma ocean model INMOM and independently, for example, to calculate tsunami waves, tides, and wind surges. The model was implemented using MPI technology, and the software architecture of the model did not involve the use of hybrid parallel programming.

The aim of this work is to create a new software architecture of the shallow water model, which involves the use of hybrid parallel programming, and to implement a hybrid MPI/OpenMP shallow water model based on this software architecture for effective use on massively parallel multiprocessor computing systems.

The model is based on a system of nonlinear shallow water equations that are solved using numerical methods. The second order numerical schemes on the ‘C’-grid according to Arakawa classification and the explicit first order „leapfrog“ scheme are used as spatial and temporal discretization schemes in the model, respectively.

A new software architecture for the shallow water model was developed, based on the separation of concerns. This software architecture divides program code into three levels. The lowest level contains all subroutines needed to calculate the nonlinear shallow water equations, the so-called computational cores. The highest level is responsible for the order of calling the computational cores and describes the time cycle of the model at a relatively high level. The intermediate level between the first two is responsible for the parallel methods and approaches used in the model. Such software architectures allow separate parallel methods and approaches from the model in order to adapt them for different types of computing systems and flexibly configure the model for the target computing system, without changing the parts of the program responsible for calculating shallow water equations.

The reported study was funded by RFBR, project number № 20-31-90109.

As the main method of parallelization, the model uses an improved two-dimensional method of domain decomposition: the initial computational domain is uniformly divided into rectangular blocks of small size and each processor is assigned a certain number of blocks, which form its computational subdomain. This approach has a number of advantages: it becomes possible to balance the load of calculations on processors and work effectively with cache memory. Due to the presence of islands and coasts in the computational area, the load balancing is a particularly urgent task. The method of load balancing using Hilbert curves is chosen as one of such methods.

In the model, a task-based hybrid MPI/OpenMP approach was implemented, in which blocks from a two-dimensional decomposition are distributed across OpenMP threads. Blocks are first distributed across MPI processes using the load balancing method, and then within the MPI process are distributed across its available threads, ensuring a uniform computational load per thread. The paper shows the advantage of this approach in comparison with the widespread vector-based MPI/OpenMP hybrid approach, in which OpenMP is used only for parallelizing two-dimensional loops over subdomains.

The paper shows the efficiency of decomposition into small blocks and the efficiency of working with cache memory when calculating the shallow water model on a single core. It was shown that the implemented task-based hybrid approach has a performance twice as high as the vector-based hybrid approach when calculating the model on many computing nodes. The advantage of the task-based hybrid approach over the pure MPI approach was shown. The efficiency of the method of load balancing was also demonstrated in the work.

Note that the implemented software architecture in the shallow water model also makes it possible in the future to apply heterogeneous parallel programming using CUDA, OpenACC, etc. All modifications during the transition to the new model of parallel programming will also be carried out without modifications of the program parts responsible for calculating the shallow water equations, which greatly simplifies the model development and support.

Key words: parallel computing, hybrid parallel programming, shallow water equations, software architecture.

References

1. Lawrence, B. N. and Rezny, M. and Budich, R. and Bauer, P. and Behrens, J. and Carter, M. And Deconinck, W. and Ford, R. and Maynard, C. and Mullerworth, S. and Osuna, C. and Porter, A. and Serradell, K. and Valcke, S. and Wedi, N. and Wilson, S. Crossing the chasm: how to develop weather and climate models for next generation computers? // Geosci. Model Dev., 2018. N 11, P. 1799–1821.
2. Volodin E. M., Dianskij N. A., Gusev A. V. Model' zemnoj sistemy INMCM4: vospriozvedenie i prognoz klimaticeskikh izmenenij v 19–21 vekah. // Izvestiya RAN. Fizika atmosfery i okeana, 2013, T. 49, N 4, S. 379–400 (in Russian).
3. Chaplygin A. V. Parallel'naya realizaciya obshchej modeli cirkulyacii okeana INMOM. // Sbornik tezisov luchshih vypusknyh kvalifikacionnyh rabot fakul'teta VMK MGU 2017. Moskva: MAKs PRESS, 2017. S. 27–28 (in Russian).
4. Chaplygin A. V., Dianskij N. A., Gusev A. V. Parallel'noe modelirovanie nelinejnyh uravnenij melkoj vody. // Trudy 60-j Vserossijskoj nauchnoj konferencii MFTI, 2017 (in Russian).
5. A.V. Chaplygin, N.A. Diansky, A. V. Gusev. Load Balancing Using Hilbert Space-Filling Curves for Parallel Shallow Water Simulations. // Numerical methods and programming. 2019. N 20. P. 75–87.
6. Arakawa A., Lamb V. R. Computational design of the basic dynamical processes of the UCLA general circulation model. // Methods in computational Physics. V. 17.
7. Mesinger F., Arakawa A. Numerical methods used in atmospheric models. // JOC, GAPR Publication Series. 1976. V. 1, N 17.

8. Rouch P. Dzh. Vychislitel'naya gidrodinamika [Computational Fluid Dynamics]. Moscow: Mir, 1980 (in Russian).
9. George L. Mellor. User guide for a three-dimensional, primitive equation, numerical ocean model. Princeton University.
10. Dianskij N. A. Modelirovanie cirkulyacii okeana i issledovanie ego reakcii na korotkoperiodnye i dolgoperiodnye atmosfernye vozdejstviya. M.: Fizmalit, 2012 (in Russian).
11. Smith R., Jones P., Briegleb B., et al. The Parallel Ocean Program (POP) Reference Manual Ocean Component of the Community Climate System Model (CCSM) and Community Earth System Model (CESM). 23 March 2010.
12. van Werkhoven B., Maassen J., Kliphuis M., Dijkstra H. A., Brunnabend S. E., van Meersbergen M., Seinstra F. J., and Bal H. E. A distributed computing approach to improve the performance of the Parallel Ocean Program. // Geosci. Model Dev., 2014. N 7, P. 267–281.
13. Wilhelmsson Tomas. Parallelization of the HIROMB Ocean Model. Licentiate Thesis, Royal Institute of Technology Department of Numerical Analysis and Computer Science, 2002.
14. John M. Dennis. Inverse Space-Filling Curve Partitioning of a Global Ocean Model. // IEEE International Parallel and Distributed Processing Symposium, 2007.
15. Liu T. et al. Parallel Implementation and Optimization of Regional Ocean Modeling System (ROMS) Based on Sunway SW26010 Many-Core Processor // IEEE Access, 2019. Vol. 7, P. 146170–146182.
16. Afzal, A., Ansari, Z., Faizabadi, A. R. et al. Parallelization Strategies for Computational Fluid Dynamics Software: State of the Art Review. // Arch Computat Methods Eng. 2017. N 24, P. 337–363.
17. Akhmetova D., Iakymchuk R., Ekeberg O., Laure E. Performance study of multithreaded MPI and Openmp tasking in a large scientific code // Proc. 2017 IEEE 31st Int. Parallel Distrib. Process. Symp. Work. IPDPSW 2017; P. 756–765.
18. Mortikov E. V. Programmnaya realizaciya bloka perenosa primej v klimaticeskikh modelyah na osnove gibridnogo programmirovaniya MPI-OpenMP // Superkomp'yuternye dni v Rossii: Trudy mezhdunarodnoj konferencii. 2016. S. 521–529 (in Russian).
19. Rabenseifner R. and Wellein G. Communication and optimization aspects of parallel programming models on hybrid architectures. // Int. J. High Perform. Comp. Appl. 2003. N 17(1), P. 49–62.
20. Elliott, S., Sobhani, N., Del Vento, D., Gill, D. O. WRF performance optimization targeting Intel multicore and manycore architectures. // In 2nd Symposium on High Performance Computing for Weather, Water, and Climate. American Meteorological Society: New Orleans, LA, US, 2016.
21. Shchepetkin Alexander F., McWilliams James C. The regional oceanic modeling system (ROMS): a split-explicit, free-surface, topography-following-coordinate oceanic model // Ocean Modelling. 2005. Vol. 9, Iss. 4, P. 347–404.
22. Andrew R. Porter, Jeremy Appleyard, Mike Ashworth, Rupert W. Ford, Jason Holt, Hedong Liu and Graham D. Riley. Portable multi- and many-core performance for finite-difference or finite-element codes — application to the free-surface component of NEMO (NEMOLite2D 1.0). // Geosci. Model Dev., 2018. N 11, P. 3447–3464.
23. NEMO Consortium. NEMO development strategy Version 2: 2018-2022. 2018.
24. Cluster INM RAS. [Electron. Res.]: <http://cluster2.inm.ras.ru>.
25. Supercomputer MVS-10P (JSCC). [Electron. Res.]: <http://www.jsc.ru>.



ГИБРИДНАЯ МОДЕЛЬ МЕЛКОЙ ВОДЫ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИЙ MPI-OPENMP

А. В. Чаплыгин, А. В. Гусев

Институт вычислительной математики им. Г. И. Марчука РАН,
119333, Москва, Россия

УДК 519.63, 519.683, 519.688

DOI: 10.24411/2073-0667-2021-10006

Гибридные модели, сочетающие в себе технологии MPI для архитектур с распределенной памятью и OpenMP для архитектур с общей памятью, становятся все более популярными, поскольку современные высокопроизводительные вычислительные системы представляют собой набор многопроцессорных систем с общей памятью, объединенных в единую коммуникационную сеть. Создание моделей, использующих эффективно ресурсы таких вычислительных систем, является актуальной задачей на сегодняшний день. В работе представлена гибридная модель мелкой воды, являющаяся одним из основных блоков сигма-модели общей циркуляции океана INMOM. Программная архитектура модели мелкой воды построена по принципу разделения обязанностей, что позволяет выделить параллельные методы и подходы в обособленную часть программы с целью их адаптации для вычислительных систем различного типа и гибкой настройки программного комплекса на целевую вычислительную систему. В модели мелкой воды был реализован гибридный подход, при котором расчетная область разбивается на блоки малого размера, которые затем распределяются по всем доступным процессам и потокам. Распределение блоков основано на методе балансировки нагрузки с использованием кривых Гильberta, что обеспечивает равномерную вычислительную нагрузку на процессы и потоки. В работе показано преимущество этого похода в сравнении с широко распространенным векторным подходом, в котором OpenMP используется только для распараллеливания двумерных циклов по подобластям. Тестирование гибридной модели проводилось на кластере ИВМ РАН и суперкомпьютере МСЦ РАН. Была показана эффективность разбиения на блоки малого размера, показана эффективность гибридного подхода в сравнении с чистым MPI режимом и также продемонстрирована эффективность метода балансировки нагрузки вычислений.

Ключевые слова: параллельные вычисления, гибридные модели параллельного программирования, уравнения мелкой воды, программная архитектура.

Введение. Гибридные модели, сочетающие в себе технологии передачи сообщений MPI для архитектур с распределенной памятью и многопоточной обработки данных OpenMP для архитектур с общей памятью, становятся все более популярными, поскольку современные высокопроизводительные вычислительные системы представляют собой набор многопроцессорных систем с общей памятью (вычислительных узлов), объединенных в единую коммуникационную сеть. Создание моделей, эффективно использующих ресурсы таких вычислительных систем, является актуальной задачей на сегодняшний день [1].

Работа выполнена в рамках научного проекта РФФИ № 20-31-90109.

Переносимость моделей на многопроцессорные системы во многом определяется правильным выбором ее программной архитектуры, которая обеспечивала бы также гибкость и простоту поддержания кода.

В работе рассматривается модель мелкой воды, являющаяся блоком сигма модели общей циркуляции океана INMOM (Institute of Numerical Mathematics Ocean Model). Модель INMOM развивается в ИВМ РАН и используется в качестве блока климатической модели INMCM (Institute of Numerical Mathematics Climate Model), созданной в ИВМ РАН и участвующей в программе IPCC (Intergovernmental Panel on Climate Change) по прогнозированию изменений климата [2]. Модель полностью написана на языке Fortran 90/95.

В работах [3, 4] была сформулирована модель мелкой воды, которая может использоваться как в качестве блока сигма модели океана INMOM, так и независимо, например, для расчетов прохождения волны цунами, приливов и ветрового нагона. В работе [4] была реализована параллельная версия модели мелкой воды для использования на вычислительных архитектурах с распределенной памятью, модель была протестирована для задач моделирования цунами в Тихом океане и шторма 2013 г. на Азовском море. В работе [5] был реализован и проанализирован метод балансировки нагрузки вычислений на процессоры с использованием фрактальных кривых Гильберта применительно к модели мелкой воды. В этих работах модель мелкой воды была реализована для использования на вычислительных системах с распределенной памятью с применением технологии MPI, и программная архитектура модели не предполагала гибкого перехода на гибридные модели параллельного программирования с совместным использованием технологий MPI и OpenMP.

Цель данной работы — создать новую программную архитектуру модели мелкой воды, предполагающую гибкий переход на гибридные модели параллельного программирования с использованием технологий MPI и OpenMP и реализовать на основе этой программной архитектуры гибридную модель мелкой воды для эффективного использования на массивно-параллельных многопроцессорных вычислительных системах.

1. Модель мелкой воды. Рассматриваемая в работе модель основана на системе нелинейных уравнений мелкой воды, которая записывается в произвольной ортогональной системе координат в следующем виде:

$$\begin{aligned} \frac{\partial r_x r_y h u}{\partial t} + T_u(u, v) - F_u(u, v) - h r_x r_y l v + r_y h g \frac{\partial \zeta}{\partial x} &= RHS_u \\ \frac{\partial r_x r_y h v}{\partial t} + T_v(u, v) - F_v(u, v) + h r_x r_y l u + r_x h g \frac{\partial \zeta}{\partial y} &= RHS_v, \\ \frac{\partial h}{\partial t} + \frac{1}{r_x r_y} \left(\frac{\partial u r_y h}{\partial x} + \frac{\partial v r_x h}{\partial y} \right) &= 0 \end{aligned} \quad (1)$$

где r_x, r_y — метрические коэффициенты Ламе, возникающие при записи системы уравнений в произвольной ортогональной системе координат; u, v — компоненты усредненного по глубине вектора горизонтальной скорости; l — параметр Кориолиса; g — ускорение свободного падения; ζ — отклонение уровня моря относительно невозмущенного состояния; $h = H + \zeta$ — полная глубина океана; H — глубина океана в состоянии покоя. Операторы переноса T_u, T_v записываются в криволинейной системе координат в дивергентной форме:

$$\begin{aligned} T_u(u, v, h) &= \frac{\partial h r_y u u}{\partial x} + \frac{\partial h r_x v v}{\partial y} - h \left(v \frac{\partial r_y}{\partial x} - u \frac{\partial r_x}{\partial y} \right) v \\ T_v(u, v, h) &= \frac{\partial h r_y u v}{\partial x} + \frac{\partial h r_x v u}{\partial y} + h \left(v \frac{\partial r_y}{\partial x} - u \frac{\partial r_x}{\partial y} \right) u. \end{aligned} \quad (2)$$

Операторы вязкости F_u , F_v записываются как дивергенция тензора напряжений:

$$\begin{aligned} F_u(u,v) &= \frac{1}{r_y} \frac{\partial}{\partial x} (r_y^2 K D_T h) + \frac{1}{r_x} \frac{\partial}{\partial y} (r_x^2 K D_S h) \\ F_v(u,v) &= -\frac{1}{r_x} \frac{\partial}{\partial y} (r_x^2 K D_T h) + \frac{1}{r_y} \frac{\partial}{\partial x} (r_y^2 K D_S h) \end{aligned} . \quad (3)$$

Здесь K — коэффициент вязкости, а D_T и D_S — компоненты тензоров напряжений сжатия-растяжения и сдвига соответственно:

$$\begin{aligned} D_T &= \frac{r_y}{r_x} \frac{\partial}{\partial x} \left(\frac{u}{r_y} \right) - \frac{r_x}{r_y} \frac{\partial}{\partial y} \left(\frac{v}{r_x} \right) \\ D_S &= \frac{r_x}{r_y} \frac{\partial}{\partial y} \left(\frac{u}{r_x} \right) + \frac{r_y}{r_x} \frac{\partial}{\partial x} \left(\frac{v}{r_y} \right) \end{aligned} . \quad (4)$$

В общем случае в правых частях RHS_u , RHS_v рассчитываются градиенты атмосферного давления и напряжения трения ветра.

На берегах для скорости задаются граничные условия непротекания и свободного скольжения.

Именно в виде (1)–(4) нелинейные уравнения мелкой воды представлены в сигма модели общей циркуляции океана INMOM, возникающие при разрешении быстрых баротропных гравитационных волн.

Система уравнений (1)–(4) в модели разрешается с использованием численных методов. При дискретизации по пространству применяется техника построения разностных аппроксимаций второго порядка точности на разнесенной 'C'-сетке по классификации Аракавы [6, 7], в общем случае используется стека нерегулярная как по долготе, так и по широте. При дискретизации по времени используется явная численная схема 'Чехарда со средней точкой' ('leapfrog'). При этом на каждом временном шаге делается фильтрация, чтобы избежать расщепления решения по нечетным и четным временным шагам, что свойственно для такой численной схемы [8, 9].

Более подробно про форму записи системы нелинейных уравнений мелкой воды и их разрешения с использованием численных методов можно посмотреть в работах [4, 5, 10].

2. Параллельная реализация для вычислительных систем с распределенной памятью. 2.1. *Метод декомпозиции подобласти.* В качестве основного метода распараллеливания для вычислительных архитектур с распределенной памятью в модели мелкой воды используется двумерный метод декомпозиции области, суть которого в следующем: исходная расчетная область разбивается на подобласти и каждому процессору ставится в соответствие своя подобласть. У каждой подобласти есть внераасчетная граница толщиной в одну точку, и если процессору нужны данные с соседней подобласти, то производится синхронизация всех процессоров, при которой заполняется внераасчетная граница каждой подобласти, и в последующих вычислениях процессор берет эти данные со своей внераасчетной границы. Синхронизация процессоров происходит с использованием технологии MPI.

Наиболее распространенным и легко реализуемым методом разбиения на подобласти является метод равномерного разбиения на прямоугольные подобласти, этот метод и механизм синхронизаций представлены на рис. 1. На рис. 1 также видно, что из-за неоднородности акватории при таком разбиении часть подобластей полностью попадает на сушу и не участвует в вычислениях, а часть загружена менее чем на половину из-за большого

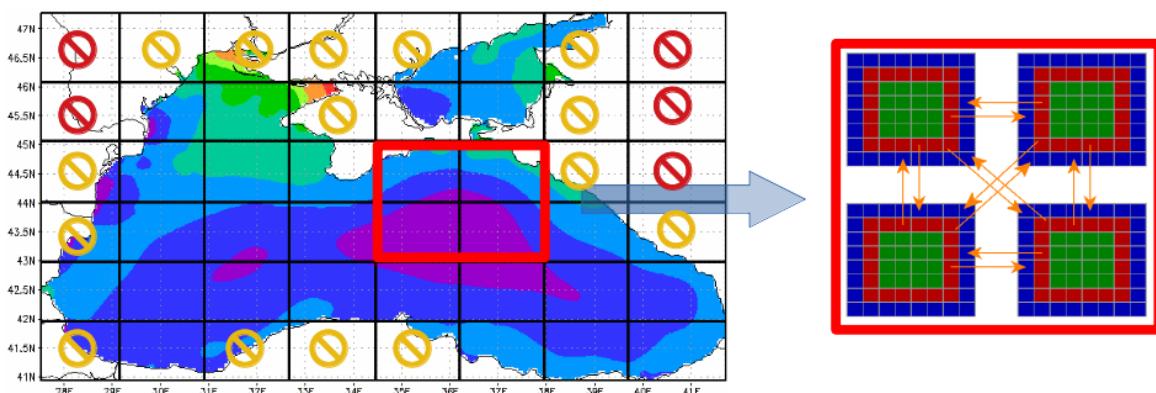


Рис. 1. Слева: равномерное разбиение на прямоугольные подобласти, красным помечены подобласти, попадающие полностью на сушу; желтым помечены подобласти, загруженные менее чем на половину. Справа: механизм синхронизаций, красным показана граница каждой подобласти, синим показана внерасчетная граница и зеленым помечены внутренние точки подобласти

количества точек суши внутри. При таком подходе выходит, что часть процессоров пропстаивает в вычислениях, а другая часть часто находится в ожидании из-за недостаточной загруженности.

Поэтому в модели был реализован усовершенствованный метод разбиения на подобласти, так называемый блочный подход. Суть этого подхода в следующем: исходная расчетная область равномерно разбивается на прямоугольные блоки малого размера, и каждому процессору ставится в соответствие некоторый набор блоков, которые и формируют его расчетную подобласть, пример показан на рис. 2. Все вычисления в модели происходят по блокам, у каждого блока есть своя внерасчетная граница. Если блоки находятся внутри одной подобласти, то при синхронизации происходит обычное копирование границы блока на внерасчетную границу соседнего блока. Если блоки находятся на соседних подобластиах, то при синхронизации используется по-прежнему технология MPI. Блочный подход уже успел зарекомендовать себя в таких моделях океана как Parallel Ocean Program (POP) [11, 12] и HIROMB (High Resolution Operational Model for the Baltic Sea) [13].

Рассматриваемый блочный подход имеет ряд преимуществ:

Подобласти могут быть произвольными многоугольниками. Действительно, т. к. подобласть состоит из набора блоков малого размера, то можно формировать подобласти довольно произвольной формы. Это свойство позволяет проводить метод балансировки нагрузки вычислений на процессоры, формируя подобласти примерно одинаковой загруженности, о чём будет более подробно написано в следующем разделе.

Эффективная работа с кэш памятью. Выбирая блок малого размера, можно получить прирост в производительности за счет эффективной работы с памятью. Для гидродинамических моделей это свойство крайне важно, т. к. любое вычисление на сетке сопровождается большим количеством обращений в память. Если блоки, для которых проводятся вычисления, помещаются полностью в кэш память, то многочисленные обращения в память перестают быть такими дорогостоящими, и можно ожидать значительный прирост в производительности.

Также блочный подход имеет недостаток: дополнительные затраты на копирование внерасчетной границы блоков при синхронизации.

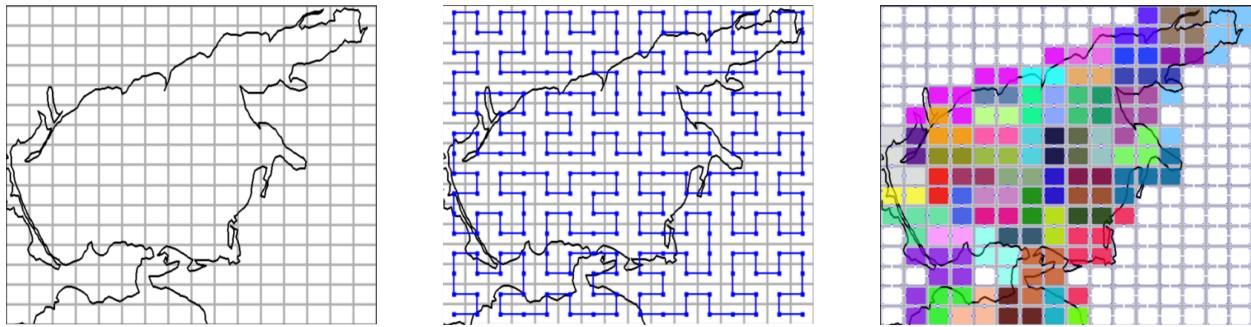


Рис. 2. Пошаговый алгоритм балансировки нагрузки с помощью кривых Гильберта

В разделе с вычислительными экспериментами будет показано, что при малом размере блоков эффективная работа с кэш памятью компенсирует затраты на копирование при синхронизации, и поэтому этот недостаток можно считать не таким существенным.

2.2. Балансировка нагрузки вычислений. Как уже было сказано ранее, блочный подход позволяет формировать подобласти в форме произвольных многоугольников, что позволяет использовать методы балансировки нагрузки вычислений при построении подобластей процессоров. В модели в качестве одного из таких методов был реализован метод балансировки нагрузки вычислений с использованием кривых Гильберта. В работах [5, 14] подробно описан и проанализирован этот метод. Здесь же кратко его опишем. Предварительно вся расчетная область равномерно разбивается на прямоугольные блоки. Для каждого блока рассчитывается значение загруженности блока как сумма всех точек, которые не лежат на сущем. Далее, на сетке блоков проводится кривая Гильберта, которая переводит двумерное пространство блоков в одномерное. Разбиение на подобласти происходит вдоль кривой и таким образом, чтобы подобласти имели примерно одинаковую сумму загруженности блоков. Затем каждому процессору ставится в соответствие его подобласть, на которой он проводит вычисления. Блоки, которые полностью состоят из точек на сущем, в распределении по процессорам и в дальнейших вычислениях не участвуют. На рис. 2 наглядно показаны шаги описанного алгоритма.

3. Гибридная модель параллельного программирования. Узкое место в рассматриваемой модели мелкой воды — это синхронизации MPI между процессорами, т. к. при увеличении числа вычислительных узлов возрастают накладные расходы на синхронизации из-за высокой нагрузки на сеть. Можно уменьшить синхронизации и тем самым уменьшить нагрузку на сеть, если использовать OpenMP для распараллеливания на общей памяти внутри узла. Это так называемый гибридный подход. Если при использовании только технологии MPI, в так называемом чистом MPI подходе, для каждого ядра на узле создается отдельный MPI процесс, то в гибридном MPI + OpenMP подходе на каждый узел создается только один MPI процесс, и для каждого ядра создаются отдельные потоки. Гибридные подходы в последнее время становятся все более актуальными и используются во многих гидродинамических моделях [15, 16, 17, 18].

Есть два подхода при использовании технологии OpenMP для распараллеливания на общей памяти [19]. Первый и наиболее распространенный подход называется векторным подходом (vector based). При этом подходе происходит распараллеливание по потокам всех вычислений в модели по подобласти, как схематично показано слева на рис. 3. Т. е. каждому MPI процессу соответствует некоторая подобласть, и все OpenMP потоки существуют

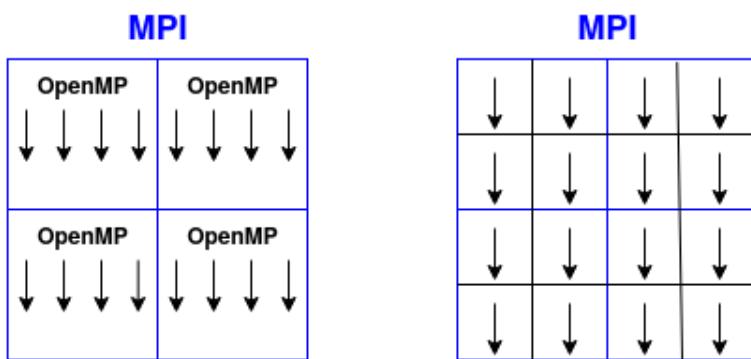


Рис. 3. Гибридные MPI + OpenMP подходы. Слева векторный подход; справа задачный подход

и проводят вычисления внутри этой общей для них подобласти. Этот подход легко реализуем, но демонстрирует малую производительность во многом из-за того, что потоки OpenMP неэффективно используют кэш память при работе на общей памяти.

Второй подход называется задачным подходом (task based). Главная идея в этом подходе заключается в том, чтобы использовать метод декомпозиции области для потоков OpenMP, т. е. ставить каждому потоку в соответствие некоторую подобласть. При таком подходе каждый поток существует и проводит вычисления на своей собственной подобласти, как схематично показано справа на рис. 3. За основу гибридного MPI + OpenMP подхода в модели мелкой воды был взят именно задачный подход. Блоки из разбиения распределяются как по процессам MPI, так и по потокам OpenMP. Распределение блоков по MPI процессам происходит с использованием метода балансировки нагрузки вычислений, описанного в предыдущем разделе, и далее происходит распределение блоков по OpenMP потокам внутри каждого MPI процесса с использованием жадного алгоритма следующим образом: все доступные блоки MPI процесса сортируются по величине загруженности и далее в этом порядке по одному распределяются на все доступные потоки внутри MPI процесса. Такой алгоритм при условии, что количество блоков в разбиении велико, обеспечивает равномерную нагрузку вычислений по всем доступным потокам в вычислительной системе.

В разделе с вычислительными экспериментами будет проведено сравнение двух описанных гибридных подходов MPI + OpenMP и продемонстрировано, что задачный подход во многом эффективнее векторного подхода. Здесь же отметим, что именно задачный OpenMP подход используется в широко известных моделях атмосферы WRF (Weather Research and Forecasting Model) [20] и океана ROMS (Regional Ocean Modeling System) [15, 21].

Отметим также следующие особенности реализованного гибридного подхода с использованием технологий MPI и OpenMP:

- У каждого процессора есть буфер для MPI обмена данными с соседними процессорами, в котором имеется место под хранение данных со всех границ блоков подобласти, необходимых для пересылки на соседние процессоры. Синхронизация между подобластями происходит с использованием этого буфера и неблокирующими вызовами MPI, причем синхронизация выполняется одним потоком.



Рис. 4. Трехуровневая программная архитектура модели мелкой воды

— При синхронизации копирование границы блоков во внерасчетную границу соседних блоков и в буфер для MPI обмена происходит параллельно всеми потоками OpenMP.

— Для OpenMP параллельный регион создается в начале запуска модели и существует до конца расчета, что минимизирует временные затраты на инициализацию потоков в модели.

— Инициализация всех данных происходит в параллельном регионе OpenMP. Это обеспечивает то, что страница памяти выделяется с узла того потока, который первый к ней обратился (first-touch policy). При таком подходе потоки эффективнее работают с памятью в NUMA-системах (Non Uniform Memory Access), в которых доступ в память чужого узла занимает существенно больше времени, чем доступ в память своего узла.

— Балансировка нагрузки вычислений по потокам происходит в начале расчета, и далее всюду используется статическое планирование OpenMP потоков по блокам, т. е. используется `schedule(static)`. При динамическом планировании, т. е. при использовании `schedule(dynamic)`, имеются накладные расходы и происходит менее эффективная работа с памятью, чем при статическом планировании.

— Всюду, где это возможно, используется `nowait` для параллельных секций OpenMP, чтобы минимизировать точки синхронизации потоков.

4. Программная архитектура. В модели была реализована программная архитектура, основанная на принципе разделения обязанностей. Программная архитектура заключается в том, что весь программный комплекс разделяется на три уровня: самый нижний уровень, уровень Ядра, содержит все процедуры, необходимые для вычисления уравнений мелкой воды, так называемые ядра модели; самый высокий уровень, уровень Алгоритма, отвечает за порядок вызова ядер модели и задает схему работы модели по времени; уровень Интерфейса выступает в качестве промежуточного уровня между первыми двумя и отвечает за параллельные методы и подходы, используемые в модели (рис. 4).

Программные архитектуры такого типа позволяют выделить параллельные методы в обособленную часть программы (уровень Интерфейса) с целью их адаптации и гибкой настройки на целевую вычислительную систему, которые происходят без изменений частей программы, отвечающих за вычисления уравнений мелкой воды. Такой подход отделяет физику модели от особенностей параллельной реализации, что упрощает поддержку и развитие всего программного комплекса.

Трехуровневая программная архитектура, основанная на принципе разделения обязанностей, зарекомендовала себя в модели мелкой воды из модели океана NEMO (Nucleus for

European Modelling of the Ocean) [22] и планируется внедрить ее в полную модель океана NEMO к 2022 году [23].

Опишем схематично каждый программный уровень модели мелкой воды.

4.1. *Уровень Ядра.* Уровень Ядра содержит все вычислительные процедуры, так называемые ядра модели. В общем случае ядро модели представляет собой некоторый двумерный цикл по прямоугольной подобласти, внутри которого происходят сеточные вычисления — это могут быть шаблоны разностных схем и прочее. На этом уровне происходит работа с обычновенными двумерными массивами, и исходная нераспараллеленная программа представляет собой набор именно таких процедур, т. е. это база модели. Всего в модели мелкой воды было выделено порядка 15 ядер. В листинге ниже представлен общий вид ядра модели, где `nx_start`, `nx_end`, ... — это границы подобласти; `bnd_x1`, `bnd_x2`, ... — это границы подобласти, включающие внерасчетную границу; `var` — сеточная переменная, которая может быть уровнем, компонентной скоростью и т. д.

```
subroutine kernel(var)
    real(wp8), intent(inout) :: var(bnd_x1:bnd_x2, bnd_y1:bnd_y2)
    ...
    do m = nx_start, nx_end
        do n = ny_start, ny_end
            var(m, n) = <some computations>
        enddo
    enddo
end subroutine
```

4.2. *Уровень Алгоритма.* На уровне Алгоритма задается порядок вызова ядер модели, тем самым на этом уровне описывается основной временной цикл модели. На этом уровне происходит работа с абстрактными структурами данных, например классом `ocean_type`, включающим в себя сеточные переменные уровня, компонент скорости и т. д. В листинге ниже схематично представлен пример кода этого уровня. На этом примере сначала вызывается ядро расчета уровня (`kernel_ssh`), затем ядро расчета компонент скорости (`kernel_uv`) и т. д. Вызовы ядер производятся с использованием специальной процедуры `envoke`, являющейся частью Интерфейса, о котором речь пойдет далее.

```
...
type(ocean_type), target :: ocean_data
procedure(empty_kernel), pointer :: kernel_ssh, kernel_uv
...
call envoke(ocean_data%ssh, kernel_ssh)
call envoke(ocean_data%u, ocean_data%v, kernel_uv)
...
```

4.3. *Уровень Интерфейса.* Промежуточный уровень между ядром и вызовом ядра — это уровень Интерфейса, в котором реализованы все параллельные методы и подходы, используемые в модели. В частности, в интерфейсе проводится разбиение на блоки (раздел 2.1), балансировка нагрузки вычислений (раздел 2.2), гибридный подход с использованием технологий MPI и OpenMP (раздел 3), синхронизации процессоров и прочее. На этом уровне происходит работа с параллельными типами данных, например классом `data2D_real8_type`, содержащим в себе распределенные данные по блокам.

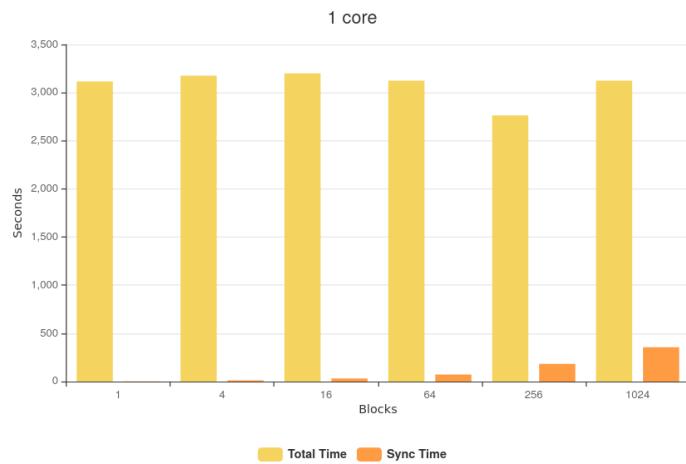


Рис. 5. Тестирование блочного подхода на одном ядре. По вертикальной оси — время в секундах: желтым цветом — общее время работы модели; оранжевым — время на синхронизации блоков. По горизонтальной оси — количество блоков в разбиении

На нижнем уровне Ядра и верхнем уровне Алгоритма содержится вся физика модели, и при этом на этих уровнях ничего не известно про особенности параллельной реализации, которые скрыты от них в Интерфейсе. Интерфейс реализуется универсальным для всех ядер модели, что позволяет гибко настраивать модель под целевую архитектуру вычислительной системы. На листинге ниже схематично представлена реализация интерфейса с помощью единой процедуры `envoke`. Эта процедура вызывает ядро для каждого блока потока OpenMP и проводит затем синхронизацию MPI процессов и потоков OpenMP.

```
subroutine envoke(var, kernel)
    type(data2D_real8_type) :: var
    ...
    !$omp do private(k) schedule(static, 1)
    do k = 1, blocks
        call kernel(var%block(k))
    enddo
    !$omp end do nowait
    call sync
end subroutine
```

5. Вычислительные эксперименты. 5.1. *Акватория без участков суши.* Первая серия вычислительных экспериментов проводилась для акватории без участков суши с размером 1525 на 1115 точек. Эта серия экспериментов проводилась с целью продемонстрировать производительность модели без эффектов неравномерной нагрузки на процессы. Расчеты проводились с шагом, по времени равным 1 секунде, всего проводилось 8640 шагов.

Был протестирован блочный подход на одном ядре, т. е. последовательная версия модели. Тестирование проводилось на процессоре Intel Xeon Silver 4214. На рис. 5 показано время работы модели в зависимости от количества блоков, участвующих в разбиении области. Видно, что с увеличением количества блоков в разбиении (т. е. с уменьшением размера

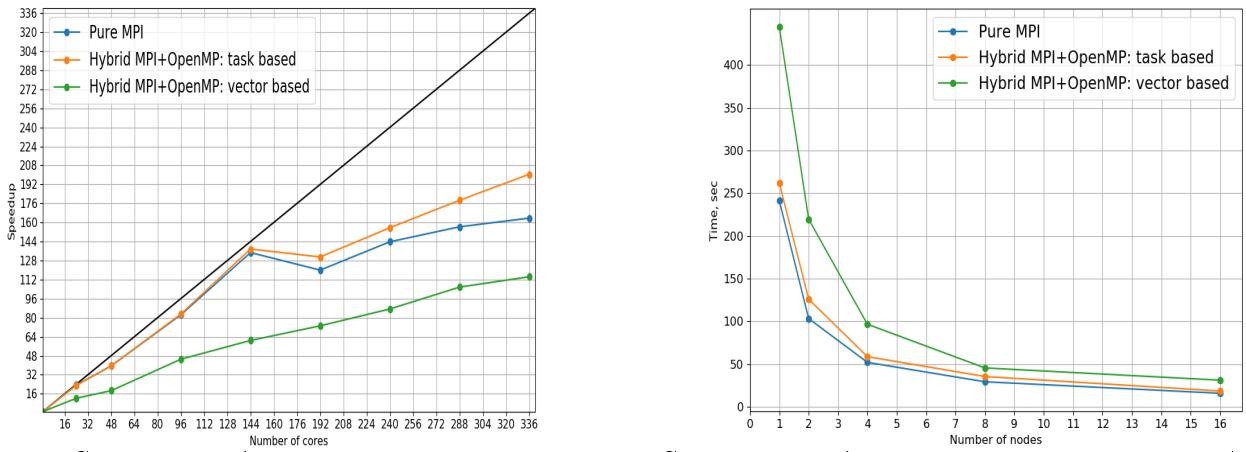


Рис. 6. Сравнение гибридных и чистого MPI подходов. Слева: масштабируемость на кластере ИВМ РАН; справа: время работы на суперкомпьютере МСЦ РАН

блока т. к. разбиение на блоки равномерное) общее время расчета модели увеличивается из-за затрат на копирование границ блоков на внерачетные границы соседних блоков. Когда размеры блоков становятся достаточно малыми, а именно когда разбиение начинает состоять из 256 блоков, происходит ускорение на 12 % по сравнению с неблочным подходом за счет эффективной работы с кэш памятью в блочном подходе. И хоть общее время расчета продолжает далее увеличиваться с уменьшением размера блоков, при 1024 блоков в разбиении оно выравнивается со временем неблочного подхода. Из всего этого можно сделать вывод, что, если выбирать малые размеры блоков, то эффективная работа с кэш памятью компенсирует затраты на копирование границ блоков при синхронизации. Для рассмотренной акватории получилось, что оптимальный размер блока, получаемый при разбиении области на 256 блоков, соответствует размеру 95 на 69 точек. Поскольку в вычислениях используются числа с плавающей запятой двойной точности (64 бита на одно число), то один такой блок занимает около 50 Кбайт в памяти.

На кластере ИВМ РАН [24] был протестирован гибридный задачный подход, и было проведено сравнение с гибридным векторным и чистым MPI подходами. Тестирование проводилось в смешанной очереди как со старыми, так и с новыми узлами. В очереди было 8 старых узлов, состоящих из двух процессоров Intel Xeon E5-2670v3 (24 ядра на узел) и 6 новых узлов, состоящих из двух более современных процессоров Intel Xeon Silver 4214 (также 24 ядра на узел). На рис. 6 показаны графики ускорения в зависимости от числа ядер, ускорение считалось относительно времени работы модели на одном ядре. На рис. 6 (слева) первые 144 ядра соответствуют 6 новым узлам очереди, и далее подключаются 8 старых узлов, максимальное количество ядер в очереди равно 336. Во всех запусках для гибридного задачного и чистого MPI подходов количество блоков бралось равному количеству ядер (т. е. каждому потоку ставился в соответствие один блок из разбиения), для гибридного векторного подхода количество блоков бралось равному количеству узлов. Из рис. 6 видно, что на 6 новых узлах гибридный задачный и чистый MPI подходы демонстрируют практически линейное ускорение, в то время как ускорение гибридного векторного подхода получается в два раза меньше. Это, как уже говорилось, во многом связано с тем, что в гибридном векторном подходе неэффективно используется кэш память потоками. При подключении старых узлов на графиках виден провал в ускорении, который связан с

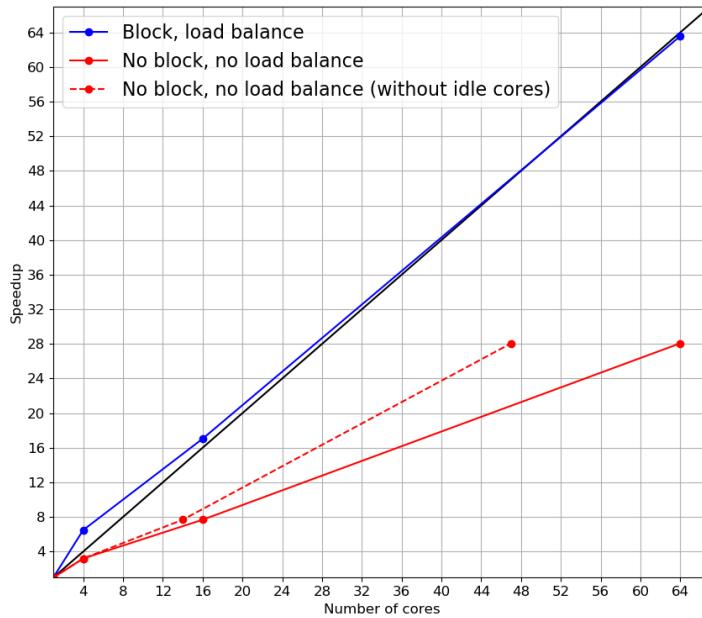


Рис. 7. Масштабируемость блочного подхода с балансировкой нагрузки вычислений на процессоры (синий) в сравнении с равномерным разбиением без балансировки нагрузки (красный)

гетерогенностью вычислений на разных процессорах. На большем количестве ядер видно, что для гибридного задачного подхода получается ускорение больше, чем в чистом MPI, из-за того, что в гибридном подходе делается меньше пересылок сообщений, чем в чистом MPI, и создается меньшая нагрузка на коммуникационную сеть.

Проводились такие же эксперименты на суперкомпьютере МВС-10П ОП1 (МСЦ РАН) на очереди broadwell, состоящей из процессоров Intel Xeon E5-2697Av4 [25]. Каждому узлу в очереди соответствуют 32 ядра, тестирование проводилось на 16 узлах (512 ядер максимально). На рис. 6 (справа) показано время работы гибридных подходов в сравнении с чистым MPI подходом. Видно, что гибридный задачный и чистый MPI подходы демонстрируют примерно одинаковое время расчета и хорошую масштабируемость, близкую к линейной. Гибридный векторный подход имеет также хорошую масштабируемость, близкую к линейной, но видно, что из-за большого времени расчета на одном узле этот метод отстает по производительности от двух других.

5.2. Акватория Азовского моря. Вторая серия экспериментов проводилась для акватории Азовского моря с разрешением 250 метров, расчетная область с размерами 1525 на 1115 точек. Расчеты проводились с шагом, по времени равным 1 секунде, всего проводилось 2160 шагов. Главная идея в этих экспериментах заключалась в том, чтобы посмотреть производительность модели мелкой воды на задаче с возможными эффектами неравномерной нагрузки на процессоры и продемонстрировать работу метода балансировки нагрузки вычислений. Акватория Азовского моря содержит довольно большое количество блоков (см. рис. 2). При разбиении этой акватории на 4096 блоков более половины блоков попадает полностью на сушу, поэтому методы балансировки нагрузки здесь будут особенно актуальными.

Вычислительные эксперименты для этой акватории проводились на кластере ИВМ РАН на 6 узлах, состоящих из двух процессоров Intel Xeon Silver 4214 (24 ядра суммарно на узел). Во всех экспериментах размер блока выбирался оптимальным для акватории, а именно выбиралось разбиение с 256 блоками при размере каждого блока, равного 95 на 69 точек (см. рис. 5). На рис. 7 показана масштабируемость блочного метода с балансировкой нагрузки (синяя линия) в сравнении с равномерным разбиением без балансировки нагрузки (красная линия). Пунктирная линия показывает ускорение, рассчитанное только для процессоров, участвующих в вычислениях, т. е. процессоры, полностью попадающие на сушу, игнорируются при подсчете ускорения. Из рис. 7 видно, что блочный метод с балансировкой нагрузки демонстрирует линейное и даже сверхлинейное ускорение на малом количестве ядер, в то время как метод без балансировки нагрузки демонстрирует ускорение в два раза меньшее. Если при подсчете ускорения игнорировать процессоры, попадающие полностью на сушу, то видно, что ускорение метода без балансировки нагрузки увеличивается примерно на 30 %, но все равно остается далеким от линейного из-за несбалансированных вычислений на загруженных процессорах.

Заключение. В работе была реализована трехуровневая программная архитектура для модели мелкой воды, основанная на принципе разделения обязанностей и выделяющая параллельные методы и подходы в обособленную часть программы с целью их адаптации и гибкой настройки под целевую вычислительную систему без изменений частей программы, отвечающих за вычисления уравнений мелкой воды. На основе этой программной архитектуры в модели мелкой воды были реализованы следующие параллельные методы и подходы:

Усовершенствованный метод декомпозиции области, так называемый блочный подход, конструирующий подобласти процессоров из набора блоков малого размера. Блочный подход позволяет создавать подобласти процессоров примерно одинаковой загруженности с использованием методов балансировки нагрузки вычислений и эффективно работать с кэш памятью при вычислениях на блоках.

Гибридный задачный подход с использованием технологий MPI и OpenMP, ставящий каждому потоку в соответствие набор блоков. В гибридном задачном подходе каждый поток существует и проводит вычисления на своей собственной подобласти, что позволяет эффективно работать потокам с кэш памятью при вычислениях на своих блоках. Гибридный подход позволяет снизить нагрузку на сеть при синхронизации внерачетных границ на большом количестве вычислительных узлов, потому что синхронизации внутри узлов не требуются из-за наличия общей памяти.

В работе было проведено тестирование модели мелкой воды: при расчете на одном ядре была показана эффективность разбиения на блоки малого размера и эффективность работы с кэш памятью; при расчете на многих вычислительных узлах было показано, что реализованный гибридный задачный подход в модели в два раза производительнее широкопространенного гибридного векторного подхода, и было показано преимущество гибридного подхода в сравнении с чистым MPI подходом. Также в работе была продемонстрирована эффективность метода балансировки нагрузки вычислений.

Отметим, что реализованная программная архитектура в модели мелкой воды позволяет осуществить гибкий переход не только на гибридные модели параллельного программирования, но и в перспективе на модели параллельного программирования для гетерогенных вычислительных систем с использованием технологий CUDA, OpenACC и т. д. Все модификации при переходе на новую модель параллельного программирования

будут осуществляться также без модификаций частей программы, отвечающих за вычисления уравнений мелкой воды, что существенно упрощает разработку и поддержку всего программного комплекса.

Список литературы

1. Lawrence, B. N. and Rezny, M. and Budich, R. and Bauer, P. and Behrens, J. and Carter, M. and Deconinck, W. and Ford, R. and Maynard, C. and Mullerworth, S. and Osuna, C. and Porter, A. and Serradell, K. and Valcke, S. and Wedi, N. and Wilson, S. Crossing the chasm: how to develop weather and climate models for next generation computers? // *Geosci. Model Dev.*, 2018. N 11. P. 1799–1821.
2. Володин Е. М., Дианский Н. А., Гусев А. В. Модель земной системы INMCM4: воспроизведение и прогноз климатических изменений в 19–21 веках // Известия РАН. Физика атмосферы и океана, 2013, Т. 49, № 4, С. 379–400.
3. Чаплыгин А. В. Параллельная реализация общей модели циркуляции океана INMOM // Сборник тезисов лучших выпускных квалификационных работ факультета ВМК МГУ 2017. Москва: МАКС ПРЕСС, 2017. С. 27–28.
4. Чаплыгин А. В., Дианский Н. А., Гусев А. В. Параллельное моделирование нелинейных уравнений мелкой воды // Труды 60-й Всероссийской научной конференции МФТИ, 2017.
5. Чаплыгин А. В., Дианский Н. А., Гусев А. В. Метод балансировки нагрузки вычислений с использованием кривых Гильберта применительно к параллельному алгоритму решения уравнений мелкой воды // Вычислительные методы и программирование. 2019. № 20. С. 75–87.
6. Arakawa A., Lamb V. R. Computational design of the basic dynamical processes of the UCLA general circulation model. // *Methods in computational Physics*. V. 17.
7. Mesinger F., Arakawa A. Numerical methods used in atmospheric models. // *JOC, GAPR Publication Series*. 1976. V. 1, N 17.
8. Роуч П. Вычислительная гидродинамика. М: Мир, 1980.
9. George L. Mellor. User guide for a three-dimensional, primitive equation, numerical ocean model. Princeton University.
10. Дианский Н. А. Моделирование циркуляции океана и исследование его реакции на короткоперiodные и долгопериодные атмосферные воздействия. М.: Физмалит, 2012.
11. Smith R., Jones P., Briegleb B., et al. The Parallel Ocean Program (POP) Reference Manual Ocean Component of the Community Climate System Model (CCSM) and Community Earth System Model (CESM). 23 March 2010.
12. van Werkhoven B., Maassen J., Kliphuis M., Dijkstra H. A., Brunnabend S. E., van Meersbergen M., Seinstra F. J., and Bal H. E. A distributed computing approach to improve the performance of the Parallel Ocean Program. // *Geosci. Model Dev.*, 2014. N 7. P. 267–281.
13. Wilhelmsson Tomas. Parallelization of the HIROMB Ocean Model. Licentiate Thesis, Royal Institute of Technology Department of Numerical Analysis and Computer Science, 2002.
14. John M. Dennis. Inverse Space-Filling Curve Partitioning of a Global Ocean Model. IEEE International Parallel and Distributed Processing Symposium, 2007.
15. Liu T. et al. Parallel Implementation and Optimization of Regional Ocean Modeling System (ROMS) Based on Sunway SW26010 Many-Core Processor. // *IEEE Access*, V. 7. P. 146170–146182, 2019.
16. Afzal, A., Ansari, Z., Faizabadi, A. R. et al. Parallelization Strategies for Computational Fluid Dynamics Software: State of the Art Review. // *Arch Computat Methods Eng* 24, 337–363 (2017).
17. Akhmetova D, Iakymchuk R, Ekeberg O, Laure E. Performance study of multithreaded MPI and Openmp tasking in a large scientific code. // Proc. 2017 IEEE 31st Int. Parallel Distrib. Process. Symp. Work. IPDPSW 2017. P. 756–65.

18. Мортиков Е. В. Программная реализация блока переноса примесей в климатических моделях на основе гибридного программирования MPI-OpenMP. // Суперкомпьютерные дни в России: Труды международной конференции. 2016. С. 521–529.
19. Rabenseifner R. and Wellein G. Communication and optimization aspects of parallel programming models on hybrid architectures. // Int. J. High Perform. Comp. Appl., 2003. N 17(1). P. 49–62.
20. Elliott, S., Sobhani, N., Del Vento, D., Gill, D. O. WRF performance optimization targeting Intel multicore and manycore architectures. In 2nd Symposium on High Performance Computing for Weather, Water, and Climate. American Meteorological Society: New Orleans, LA, US, 2016.
21. Shchepetkin Alexander F., McWilliams James C., The regional oceanic modeling system (ROMS): a split-explicit, free-surface, topography-following-coordinate oceanic model // Ocean Modelling. 2005. V. 9. Iss. 4. P. 347–404.
22. Porter Andrew R., Appleyard Jeremy, Ashworth Mike, Ford Rupert W., Holt Jason, Liu Hedong and Riley Graham D.. Portable multi- and many-core performance for finite-difference or finite-element codes — application to the free-surface component of NEMO (NEMOLite2D 1.0). // Geosci. Model Dev., 2018. N 11. P. 3447–3464.
23. NEMO Consortium. NEMO development strategy Version 2: 2018-2022. 2018.
24. Кластер ИВМ РАН. [Электрон. ресурс]: <http://cluster2.inm.ras.ru>.
25. Кластер МСЦ РАН. [Электрон. ресурс]: <http://www.jscc.ru>.



Чаплыгин А. В. — e-mail: achaplygin99@gmail.com, тел. +7 965 1818175. Окончил бакалавриат ВМК МГУ по направлению 01.03.02 „Прикладная математика и информатика“, тема диплома: „Реализация параллельной версии модели циркуляции океана INMOM“.

С этим дипломом он победил в конкурсе выпускных квалификационных работ 2017 года факультета ВМК МГУ, получил диплом победителя 2 степени. В 2019 году с отличием окончил магистратуру ВМК МГУ по направлению 01.04.02 „Прикладная математика и информатика“, тема магистерской диссертации: „Метод балансировки нагрузки вычислений с использованием фрактальных кривых Гильберта для модели INMOM“. Два раза участвовал в международной Римско-Московской школе по матричным методам и прикладной линейной алгебре, а также проходил длительные стажировки в качестве программиста-исследователя в высокотехнологичных компаниях, где работал с высокопроизводительными программными комплексами по численному моделированию. В настоящее время является аспирантом ИВМ РАН. Область научной деятельности связана с математическим моделированием, па-

раллельным программированием и численными методами. Чаплыгин А. В. имеет 4 публикации в научных изданиях, индексируемых в РИНЦ и входящих в текущий перечень ВАК России, из них 2 публикации из WoS и Scopus; 5 публикаций в материалах конференций, индексируемых в РИНЦ.

Chaplygin A. V. graduated from the Bachelor's degree program of the Moscow State University in the direction 01.03.02 „Applied Mathematics and Computer Science“. In 2019, he graduated with honors from the Master's program of the Moscow State University in the direction 01.04.02 „Applied Mathematics and Computer Science“. He participated twice in the international Roman-Moscow School of Matrix Methods and Applied linear algebra, and also completed long-term internships as a research programmer in high-tech companies, where he worked with high-performance software for numerical modeling. Currently, he is a post-graduate student of the Marchuk Institute of Numerical Mathematics. The field of scientific activity of Chaplygin A.V. is connected with mathematical modeling, parallel programming and numerical methods. Chaplygin A.V. has 4 scientific publications, including 2 publications from WoS and Scopus; 5 publications in conference materials.



Гусев А. В. (индекс хирша 17 по РИНЦ и 12 по WoS, за последние 5 лет 50 публикаций, из них 19 из WoS и Scopus) старш. науч. сотр. ИВМ РАН и ИО РАН, канд. физ.-мат. наук. e-mail: anatoly.v.gusev@gmail.com, тел. +7 903 2467187.

Окончил Московский физико-технический институт в 2005 году, после чего был принят в аспирантуру Института вычислительной математики РАН. В 2009 году защитил диссертацию на соискание ученой степени кандидата физико-математических наук по теме „Численная модель гидродинамики океана в криволинейных координатах для воспроизведения циркуляции Мирового океана и его отдельных акваторий“ по специальности 05.13.18 — „Математическое моделирование, численные методы и комплексы программ“. В настоящее время работает в ИВМ РАН на должности старшего научного сотрудника. Область научной деятельности Гусева А. В. связана с численным моделированием циркуляции океанов и морей. В основе исследований, проводимых в ИВМ РАН по этому направлению, лежит научная база, создаваемая на протяжении многих лет основателем института академиком Г. И. Марчуком и его учениками. За время работы Гусев А. В. внес большой вклад в создание и развитие числен-

ной модели циркуляции Мирового океана ИВМ РАН и модели климатической системы Земли. Разработанная им модель, известная в международном сообществе как INMOM (Institute of Numerical Mathematics Ocean Model), находит широкое применение как в России, так и за рубежом.

Gusev A. V. (Hirsch index 17 in the RSCI and 12 in WoS, 50 publications over the past 5 years, 19 of them from WoS and Scopus) Ph. D. He graduated from the Moscow Institute of Physics and Technology in 2005, after which he was accepted to the postgraduate program of the Institute of Numerical Mathematics of the Russian Academy of Sciences (INM RAS). In 2009, he defended his Ph. D. thesis in the specialty 05.13.18 „Mathematical modeling, numerical methods and software packages“. Currently, he works at the INM RAS as a senior researcher. The field of scientific activity of Gusev A.V. is connected with numerical ocean circulation modeling. During his work, Gusev A. V. made a great contribution to the creation and development of the numerical world ocean circulation model and the model of the Earth's climate system. The model developed by him, known in the international community as the INMOM (Institute of Numerical Mathematics Ocean Model), is widely used both in Russia and abroad.

Дата поступления — 03.02.2021

Правила представления и подготовки рукописей для публикации в журнале „ПРОБЛЕМЫ ИНФОРМАТИКИ“

Общие требования.

Редакция принимает к рассмотрению статьи в электронном виде (*исходный файл в LATEX и файл PDF, либо MS Word с приложением оригиналами рисунков в формате тех программ, в которых они были сделаны, отдельными файлами*).

Файлы, содержащие текст статьи, иллюстрации и дополнительные материалы, можно пересыпать на электронный адрес редакции: problem-info@sscc.ru.

Принимаются файлы, архивированные архиваторами ZIP/7Z или RAR; применение самораспаковывающихся архивов не допускается.

При повторной отправке материалов, а также при внесении в исходный текст дополнений или исправлений необходимо сообщить об этом в редакцию в тексте электронного письма.

Текст статьи с формулами, рисунками, таблицами должен быть подготовлен на стандартном листе формата А4 через 1,5 интервала, размер шрифта 12 pt, все поля по 2 см.

Статьи, содержащие формулы, следует набирать в редакторе LATEX. В остальных случаях допускается использование программы MSWord, шрифт — TimesNewRoman; автоматическая расстановка переносов в документе должна быть отключена.

Направляя статью в редакцию журнала, автор (соавторы) на безвозмездной основе передает(ют) изда-телю на срок действия авторского права по действующему законодательству РФ исключительное право на использование статьи или отдельной ее части (в случае принятия редколлегией Журнала статьи к опубликованию) на территории всех государств, где авторские права в силу международных договоров Российской Федерации являются охраняемыми, в том числе следующие права: на воспроизведение, на распространение, на публичный показ, на доведение до всеобщего сведения, на перевод на иностранные языки и переработку (и исключительное право на использование переведенного и (или) переработанного произведения вышеуказанными способами), на предоставление всех вышеперечисленных прав другим лицам.

Журнал „Проблемы информатики“ является некоммерческим изданием. Плата с авторов за публикацию статей не взимается.

К статье должны быть приложены:

— разрешение на публикацию от экспертного совета организации, в которой выполнена работа (для авторов из России);

— оригинал рецензии;

портретные фотографии авторов разрешением не менее 300 dpi.

— Блоки информации и на русском, и на английском языках просьба присыпать отдельными файлами:

— Название статьи;

— Инициалы и фамилии авторов;

— Места работы авторов: полное наименование организации, почтовый индекс, город, страна;

— Код(ы) классификации УДК;

— Аннотации, содержащие краткую постановку задачи и описание метода решения: на русском языке объемом не более 1000 знаков, на английском языке расширенную, объемом от 4000 до 8000 знаков, что соответствует требованиям ВАК и Scopus.

— Ключевые слова;

— Списки используемой литературы в соответствии с ГОСТ Р 7.0.5—2008 (в английской версии необходимо выполнить транслитерацию неанглоязычных элементов списка литературы в соответствии с ГОСТ Р 7.0.34-2014) — составляются по ходу упоминания источников в тексте;

— Краткие биографии (БИО) авторов с указанием ключевых научных достижений (включая учченую степень, ученое звание — при наличии; основные области научных интересов и формулировку основных результатов, место работы, занимаемую должность, контактные данные — почтовый адрес с индексом, адрес электронной почты, контактный телефон).

Подготовка статьи.

1. **Материал** статьи должен быть изложен в следующей последовательности:

1. название статьи на английском языке;
2. инициалы и фамилия автора(ов) на английском языке;
3. место работы автора(ов) (на английском языке): полное наименование организации, индекс, город, страна;
4. англоязычная аннотация;

5. ключевые слова на английском языке;
6. references+транслитерация неанглоязычных элементов списка литературы;
7. название статьи на русском языке;
8. инициалы и фамилии и авторов;
9. место работы авторов: полное наименование организации, почтовый индекс, город, страна;
10. индекс УДК;
11. аннотация на русском языке;
12. ключевые слова (не более 8);
13. текст статьи;
14. список литературы, оформленный в соответствии с требованиями ГОСТ;
15. краткие биографии авторов на английском и русском языках с указанием ключевых научных достижений (ученую степень, ученое звание — при наличии; место работы, занимаемую должность, контактные данные — почтовый адрес с индексом, адрес электронной почты, контактный телефон, основные области научных интересов и формулировка основных результатов).

2. Требования к формулам:

- Нумерация формул сквозная, выносные формулы центрируются, номер выровнен по правому краю.

3. Требования к рисункам:

- Файлы с рисунками присылаются отдельно в формате программ, в которых они были выполнены: в формате MS Excel (для графиков и диаграмм), eps, pdf, png, tiff, bmp или jpg (с максимальным качеством).

- Рисунки с подрисуточными подписями заверстываются в текст статьи.

- Тексты, являющиеся частью рисунка, выполняются шрифтом TimesNewRoman.

- Фотографии должны иметь разрешение не менее 300 dpi.

4. Дополнительные требования:

- В текст статьи необходимо включать ссылки на рисунки и таблицы, а также подрисуточные подписи и заголовки таблиц. Все буквенные обозначения, приведенные на рисунках, необходимо пояснить в основном тексте или в подрисуточных подписях.

- Сокращения слов не допускаются (кроме общепринятых).

- Векторные переменные обозначаются полужирным шрифтом без курсива.

- Таблицы не должны быть громоздкими. Значения физических величин в таблицах, на графиках и в тексте должны указываться в единицах измерения СИ.

- Графики, если их на рисунке несколько, а также отдельные детали на чертежах, узлы и линии на схемах следует обозначать цифрами, набранными курсивом.

- Нумеровать следует только те формулы и уравнения, на которые имеются ссылки в тексте, нумерация сквозная.

- Ссылки на источники в тексте заключаются в квадратные скобки.

- Иностранные источники приводятся на языке оригинала. Ссылки на неопубликованные работы не допускаются.

Все статьи, опубликованные в журнале „Проблемы информатики“, доступны на сайте https://elibrary.ru/title_about.asp?id=30275 и на сайте журнала <http://problem-info.sscru> спустя год после опубликования.

Пример оформления статей можно посмотреть на сайте журнала <http://problem-info.sscru>.