МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Механико-математический факультет

ДЕВЯТАЯ СИБИРСКАЯ КОНФЕРЕНЦИЯ
ПО ПАРАЛЛЕЛЬНЫМ И ВЫСОКОПРОИЗВОДИТЕЛЬНЫМ ВЫЧИСЛЕНИЯМ

Томск, 10–12 октября 2017 года

Сборник статей

Под редакцией
д-ра физ.-мат. наук, профессора А.В. Старченко

Томск
Издательский Дом Томского государственного университета
2017

УДК 519.6
ББК 22.19
Д25

Д25 **Девятая** Сибирская конференция по параллельным и высокопроизводительным вычислениям : сборник статей / под ред. А.В. Старченко. – Томск : Издательский Дом Томского государственного университета. – 110 с.

ISBN 978-5-94621-653-1

Включены материалы Девятой Сибирской конференции по параллельным и высокопроизводительным вычислениям, проходившей 10–12 октября 2017 г. в Томском государственном университете при поддержке Министерства образования и науки РФ, Суперкомпьютерного консорциума России и программы повышения конкурентоспособности ТГУ.

Рассмотрены актуальные проблемы организации параллельных вычислений на многопроцессорных системах, современное состояние и перспективы развития методов параллельных вычислений.

Для студентов, аспирантов, преподавателей, научных работников, желающих изучить и практически использовать в научной работе высокопроизводительные вычислительные ресурсы.

УДК 519.6
ББК 22.19

ISBN 978-5-94621-653-1 © Томский государственный университет, 2017
© Авторы статей, 2017

СОДЕРЖАНИЕ

Shahinyan A.A., Astsatryan H.V., Poghosyan A.H. Green computing in molecular dynamics simulations 5

Смолехо И.В., Садовская О.В. Применение технологии CUDA при численном моделировании поведения жидких кристаллов 13

Данилкин Е.А., Грудович Л.Е., Лещинский Д.В. Параллельная реализация микромасштабной математической модели движения воздушных масс и переноса примеси 21

Гладких В.С., Пудов С.Г. Использование векторизованного формата хранения разреженных матриц для увеличения скорости матрично-векторного умножения 28

Зюзьков В.М. Вычисление сильно-составных чисел 35

Лещинский Д.В., Данилкин Е.А. Распараллеливание явных и не явных разностных схем на примере численного решения уравнения теплопроводности 42

Фефелов Н.П. Опыт освоения технологий параллельного программирования в Томском университете систем управления и радиоэлектроники (ТУСУР) 50

Потоцкая А.А., Захарова А.А. Обработка изображений с помощью Shearlets 55

Чуруксаева В.В., Старченко А.В. Численное исследование течения в русле реки во время весеннего ледохода для определения возможных зон локальных подтоплений 62

Старченко А.В., Данилкин Е.А., Семёнова А.А., Лещинский Д.В. Численное моделирование турбулентного течения в уличном каньоне при смешанной конвекции 70

Чу А.Р., Михайлов М.Д. Исследование математической модели «хищник–жертва» с учётом внутривидовой конкуренции и ареала обитания в двумерном приближении 78

Крутиков В.А., Ботыгин И.А., Шерстнёв В.С. Программный комплекс для построения распределенной вычислительной системы со слабо связанными совокупностями вычислителей 86

Волков Ю.В., Ботыгин И.А., Нагиев А.Е. Разработка и исследование алгоритма балансировки нагрузки на основе обобщенной характеристики вычислителей 92

Проханов С.А., Шумилов Б.М., Кудуев А.Ж., Абдыкалыков К.А. Использование алгоритмов обработки данных мобильных видеоизмерений для распознавания и классификации повреждений дорожной поверхности 98

Тартаковский В.А., Ботыгин И.А., Шерстнёва А.И. Разработка программных средств для обработки больших временных рядов 105

DOI: 10.17223/9785946216531/1

Green computing in molecular dynamics simulations*

A.A. Shahinyan, H.V. Astsatryan, A.H. Poghosyan

The International Scientific-Educational Center of NAS RA, Yerevan, Armenia
artsha@sci.am

One of the main problems in the Molecular Dynamics (MD) is the optimization of energy consumption and the integration of computing performance with energy efficiency. The graphical processor units (GPU) are intensively used to increase computational performance and what is most important, the usage of of many HPC applications. A series of benchmarking simulations were carried out using the GROMACS software package to identify the Tesla K40 graphical card at various frequencies to estimate the energy efficiency depending on frequency. Analysis of the data shows that the performance of the Tesla K40 is equivalent to the performance of 1024 cores of the IBM BlueGene/P or the HP CP4000BL supercomputer. Thus, it can be argued that optimal energy efficiency is achieved by using low-frequency GPUs. In this case, the use of low frequencies leads to a reduction in energy consumption by 20-30%.
Keywords: Molecular dynamics simulation, GROMACS; GPU; energy-efficient computing.

Introduction

Green computing or Energy efficiency are terms indicate the environmentally responsible use of computing recourses and any other technologies connected with them (network devices, storages etc). It is an implementation of practices, methods and different approaches to reduce power consumption without affecting the computational performance, taking into account different packages and different environments.

Recently, a series of emerging low-power graphical units (GPUs) have been announced by Nvidia Inc.* aimed to accelerating scientific computing [1]. Nvidia Tesla professional series provide ultra-fast computations, compatibility and reliability for many applications. It is known that GPU consumes much more power than a CPU, however the total amount of watt consumed per unit gigahertz is much lower when compared to CPU [2]. The CPU/GPU

* The work presented in this paper was supported by EC through the VI-SEEM Project (Nr. 675121).

hybrid calculations lead to the improving the performance and what is most important to the energy efficiency. The latter is ongoing challenge in HPC namely from economical point of view. Dynamic voltage and frequency scaling (DVFS) is a power-management technique where the processor frequency is decreased/increased leading to a change in the supply voltage [3], which makes it possible to improve the performance by switching/adapting the frequency to better suit job characteristics.

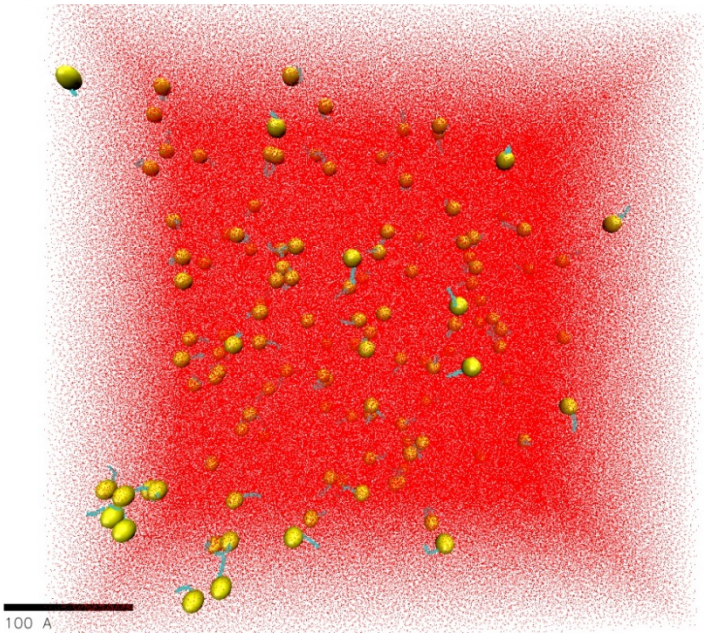


Fig. 1. Snapshot extracted from the beginning of simulation run for 128 SPDS/ water. Atoms are represented using the following colors: sulfurs – yellow, carbons – cyan. Water are colored red (points). The snapshots were rendered using VMD

The mentioned resources are used for molecular dynamics (MD) simulations. The free-available software codes, such as GROMACS [4] and NAMD [5] are commonly used, which are developed namely on GPUs. Note that the GROMACS package benchmarking was intensively done on different platforms [6].

In this paper, we try to investigate the effects of GPU DVFS on the GROMACS package performance. Using a set of testing systems, we deter-

mine the simulation performance with and without GPU acceleration by changing processor clock from 666MHz to boosting 875MHz. During the testing simulation, we pay attention to power consumption problem. The GPU coming results were compared with two different HPC platforms (Bulgarian IBM Blue Gene/P and Hungarian a fatnode HP cluster).

All benchmarking experiments have been carried out using the HP server consists of 12 dual-core Xeon E52620 processors and Nvidia Tesla K40 Kepler GPU graphical card plugged into a PCI-e slot and 65GB system memory. The OS of the server is Ubuntu 14.04, the NVIDIA System Management Interface (nividia-smi) was used for changing the GPU frequency, and also to obtain the average power consumption in watt.

Methodology

The benchmarking data, including the statistic runs, have been carried out using GROMACS 5.0.4 software package Ubuntu Linux OS with standard compilation and optimization options. All the systems were run for 10000 steps, by changing the number of processors from 1 to 24 using GPU. The benchmarking system consisting about ~2.5M atoms, have been used for the simulations with the following parameters: 2fs time step, the PME electrostatics, van der Waals forces truncated at 0.9nm with the corresponding temperature (v-rescale algorithm) and pressure control (Berendsen algorithm). Note that the runs were typically for 10000 steps without any writing outputs. Mdrun –pin flag was on, which set affinities for a cores, i. e. the threads remain locked to cores and do not jump around.

It should be noted that studied system, consisting of 2595136 atoms, is a randomly distributed 128 sodium pentadecyl sulfonate (SPDS) molecules in aqueous solution consisting of 864192 water molecules (~36x36x20nm3 cell size) (see Figure 1). The aim of this work is to better understand the self-organization features of surfactant at the concentrations near critical micelle concentration (CMC). Simultaneously with ~2.5M system, we built two other systems with SPDS molecules by increasing the number of water molecules in order to get larger systems. Corresponding systems are: ~5M atoms, consisting of 128 SPDS molecules in water, where the number of water molecule in ~37x37x37nm3 cell, are 1687840 and ~10M atoms, which contains 128 SPDS and 3388396 water molecules in ~47x47x47nm3 cell. Note that only ~2.5M system was subjected to frequency change analysis. It should be noted that further increase in size leads to the memory dump, as we have received a giga-

byte of input/output data (a coordinate GROMACS .gro file of ~100M atoms was ~4.5GB).

The following platforms were used to do benchmarking:

12 dual-core Xeon E5-2620 processors, Nvidia Tesla K40 Kepler GPU graphical card, where each core of the processors has cache size (Level 1) 6 x 32 KB and total of 2880 CUDA cores.

Bulgarian IBM Blue Gene/P supercomputer, two racks, 2048 PowerPC 450 based compute nodes, 8192 processor cores and a total of 4 TB random access memory (running 27 trillion operations a second).

The Hungarian a fatnode HP cluster based on blade technology CP4000BL and consists of 2304 cores (14 Tflops). The processors are AMD Opteron 6174 type units with 12-core Magny-Cours (2.1GHz). The network is the QDR 4x Infiniband internal high-performance communication.

The studied system is available at http://www.bioinformatics.am.

Results and discussions

The benchmarking results for system with 2.5K atoms are represented in Figure 2, *a*. The parameter describing the speed of calculation is nanoseconds per day. We have done a testing for 1, 4, 8, 16 and 24 cores and GPU, where the mdrun option was: mdrun -pin on -nb cpu -nt number-of-cores.

The data in Figure 2, *a* claims that in case of simulations without GPU, we see gradually increase of performance with the increase of processors/cores, however, GPU acceleration leads to the cardinally increase of speedup, i.e. from 3 to 4 times. Even for such kind of huge system, we track the performance up to 0.77ns/day. The best data for a mentioned system is achieved when the frequency is 875MHz with 24CPU/1GPU set. To well understand the influence of system size on performance, we have checked the largest systems (~5M and ~10M atoms) on GPU. The corresponding plot is shown in Figure 2. *b*. For two systems, we track the gradually increase in performance with the increase of processors/threads. We argue that using GPU makes it possible to scale linear for the huge systems with up 10milion particles.

To check GPU speed, we compare GPU benchmarking data with HPC resources. As a system, we use ~2.5M system for comparision. The comparison of GPU data with other HPC platforms are provided in Table.

The comparison of all data claims that the server with Tesla K40 can run a simulation as fast as on 1024 cores of IBM BlueGene/P supercomputer and more than 64 CPUs of HP CP4000BL cluster in case of the huge system. Note

that further increase of processors in case of supermachine, we track an increase of performance. Thus, we argue that 24 cores with GPU support even with low frequency show same results as 1024 cores of BlueGene/P machine and 64 cores of modern HP cluster, which is in turn, shows the great effectiveness from energy consumption point of view.

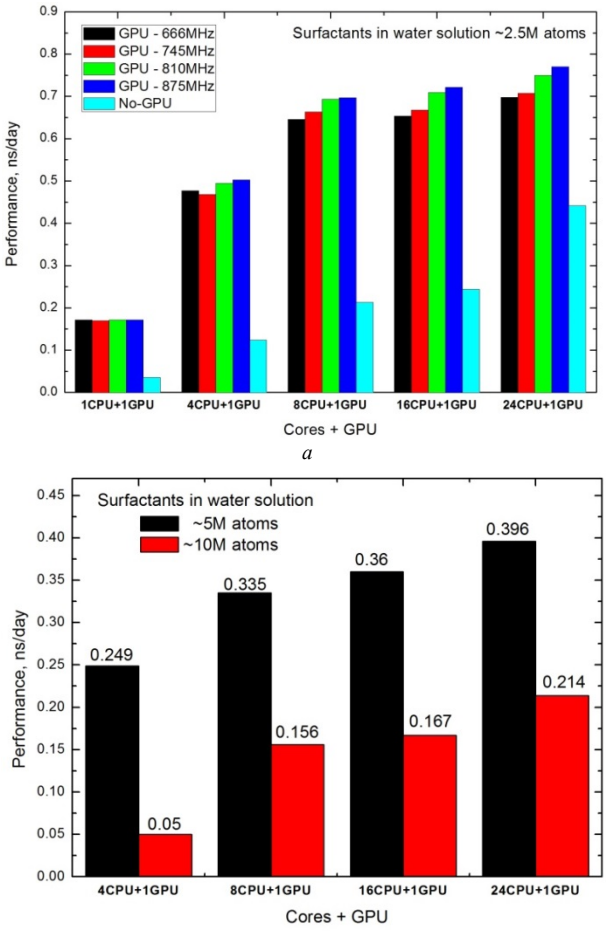


Fig. 2. The performance of system with ~2.5M (*a*) and 5M, 10M (*b*) atoms depending on number of cores and GPU

The benchmarking for a system with ~2.5M atoms. The data were extracted from [6]

Bulgarian IBM BlueGene/P		Hungarian HP CP4000BL CP4000BL		12DC Xeon E5-2620 with K40				
Cores	Results, ns/day	Cores	Results, ns/day	Cores	Results, ns/day			
128	0.132	16	0.149	4	0.124 0.213 0.244 0.442			
256	0.237	32	0.292	8	0.213			
1024	0.515	48	0.435	16	0.244			
2048	1.59	64	0.564	24	0.442			
		96	0.827	*Freq. MHz*	666	745	810	875
		128	1.1	4CPU+1GPU	0.477	0.468	0.495	0.503
		256	2.035	8CPU+1GPU	0.646	0.663	0.693	0.696
		512	3.655	16CPU+1GPU	0.654	0.668	0.709	0.722
				24CPU+1GP	0.698	0.707	0.75	0.77

To check the power consumption, we measure also the average, maximum and minimum power using nvidia-smi (Nvidia System Management Interface) module with –q –d tags (the outputs are in watts).

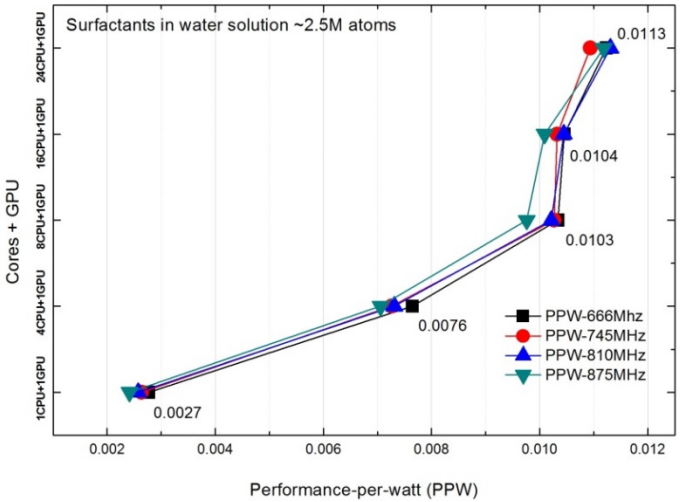


Fig. 3. The performance-per-watt of system with ~2.5M atoms depending on number of cores and GPU

The corresponding results are shown in Figure 3, where parameter called «performance-per-watt (PPW)» has been utilized. Note that the PPW parameter means the performance divided by average power. Analyzing the PPW for a huge system, we argue that optimal ratio is done for low frequency GPU (see Figure 3).

We have also estimated the energy power (wattage) used by both cases based on theoretical assumptions. Let's consider that both machines are in 80% used and 20% idle, then, on BlueGene/P machine, we will get (1024 nodes x 0.8) x (90 days //duration of current experiment on BlueGene/P) x (24 hours/day) ~ 18x105 nodes hours → (18x105 core hours x 200 Watt /node) / 100000 ~350MW hours (at 0.515 ns/day). The similar calculations in GPU machine will lead to following value - (8 nodes + 1GPU x 0.8) x (~70 days //duration of current experiment on GPU) x (24 hours/day) ~ 12x103 nodes hours → (12x103 core hours x 150 Watt /node) / 100000 ~1.8MW hours (at 0.646 ns/day), i.e. in case of graphical processors, we have ~200 times better result in term of energetical view.

Conclusion

We have done a number of statistical benchmarking runs to check the performance of GROMACS software package on GPU equipped server. The current runs have been performed to testify Tesla K40 graphical cards and estimate the consumed energy.

We showed that the GROMACS performance depends on the size of system studied and scales well even with huge number of atoms. We strongly recommend use GPU with low frequency.

The comparison of data claims that workstations with Tesla K40 can run a simulation as fast as on 1024 cores of IBM BlueGene/P supercomputer and more than 64 CPUs of HP CP4000BL cluster. We argue that 24 cores with GPU support even with low frequency show same results as 1024 cores of BlueGene/P machine and 64 cores of modern HP cluster, which is in turn, shows the great effectiveness from energy consumption point of view.

It is stated that the optimal power efficiency is achieved with the low frequency GPU. And there is a potential to have better energy efficiency by designing effective DVFS schedulers for such systems.

References

1. Nickolls J., Buck I., Garland M. and Skadron K. Scalable Parallel Programming with CUDA. ACM Queue, 2008. No. 6(2). Pp. 40–53.

2. Krishnan Y. N., Dwivedi V. and Bhagwat C. N. Green computing using graphical processing units. International Journal of Computer Applications, 2012. No 44 (19). Pp. 1–3.
3. Mittal S. A survey of techniques for improving energy efficiency in embedded computing systems. IJCAET, 2014. No 6(4). Pp. 440–459.
4. van der Spoel D., Buuren A., Apol E., Meulenhoff P., Tieleman D., A. Sijbers, Hess B., Feenstra K., Lindahl E., van Drunen R. and Berendsen H. Gromacs User Manual Version. Groningen, University of Groningen, 2002.
5. Kale L., Skeel R., Bhandarkar M., Brunner R., Gursoy A., Krawetz N., Phillips J., Shinozaki A., Varadarajan K. and Schulten K. NAMD2: Greater Scalability for Parallel Molecular Dynamics. Journal of Computational Physics, 1999. No 151(1). Pp. 283–312.
6. Poghosyan A.H., Astsatryan H.V. and Shahinyan A.A. Parallel Peculiarities and Performance of GROMACS Package on HPC Platforms.. International Journal of Scientific and Engineering Research, 2013. No 4(12). Pp. 1755–1761.

DOI: 10.17223/9785946216531/2

Применение технологии CUDA при численном моделировании поведения жидких кристаллов*

И.В. Смолехо, О.В. Садовская

Институт вычислительного моделирования СО РАН, Красноярск
ismol@icm.krasn.ru, o_sadov@icm.krasn.ru

Для описания термомеханических процессов в жидких кристаллах используется математическая модель, в рамках которой жидкокристаллическая среда рассматривается как мелкодисперсная сплошная среда с вращающимися частицами. Разработаны вычислительные алгоритмы для решения системы уравнений модели и уравнений второго порядка для касательного напряжения и угловой скорости, полученных из этой системы. Учитываются моментные взаимодействия частиц, вызванные термомеханическими возмущениями и воздействием электрических полей на жидкие кристаллы. Алгоритмы реализованы в виде параллельных программ, написанных на языке Си с применением технологии CUDA.

Ключевые слова: жидкий кристалл, динамика, упругость, теплопроводность, электрическое поле, параллельный алгоритм, технология CUDA.

Введение

Жидкий кристалл – фазовое состояние, существующее в интервале от температуры плавления кристаллов некоторых веществ до более высокой температуры, при нагреве до которой жидкий кристалл переходит в обычную жидкость. Наиболее характерным свойством жидких кристаллов является их способность изменять ориентацию молекул под воздействием электрических полей, что открывает широкое применение их в промышленности. Физические свойства жидких кристаллов описаны в [1].

В данной работе описывается алгоритм численной реализации математической модели жидкого кристалла. Разработанный ранее алгоритм усложняется с целью учета моментных взаимодействий частиц и воздействий электрического поля на жидкие кристаллы. Отдельно исследуется подсистема уравнений второго порядка для касательного напряжения и угловой скорости, полученная из основных уравнений модели. Созданы параллельные программы с применением технологии CUDA, проведены тестовые расчеты.

* Работа выполнена при финансовой поддержке Комплексной программы фундаментальных исследований СО РАН № II.2П «Интеграция и развитие» (проект № 0356-2016-0728).

Математическая модель жидкого кристалла

В работе [2] построена упрощенная математическая модель нематического жидкого кристалла как акустической микронеоднородной среды с вращающимися частицами без учета моментных свойств. Основная система уравнений этой модели в двумерном случае с учетом моментных взаимодействий выглядит так:

$$
\begin{gathered}
\rho v_{1,t} = -p_{,x} - q_{,y} + f_1, \quad \rho v_{2,t} = q_{,x} - p_{,y} + f_2, \\
j\,\omega_{,t} = 2q + \mu_{1,x} + \mu_{2,y} + m, \quad \varphi_{,t} = \omega, \quad \mu_{1,t} = \gamma\,\omega_{,x}, \quad \mu_{2,t} = \gamma\,\omega_{,y}, \\
p_{,t} = -\kappa\left(v_{1,x} + v_{2,y}\right) + \beta T_{,t}, \quad q_{,t} = \alpha\left(v_{2,x} - v_{1,y}\right) - 2\alpha\left(\omega + q/\eta\right), \\
cT_{,t} = \left(æ_{11}\,T_{,x} + æ_{12}\,T_{,y}\right)_{,x} + \left(æ_{12}\,T_{,x} + æ_{22}\,T_{,y}\right)_{,y} - \\
- \beta T\left(v_{1,x} + v_{2,y}\right) + 2q^2/\eta + H,
\end{gathered}
\qquad (1)
$$

где v_1 и v_2 – компоненты вектора скорости, ω – угловая скорость, φ – угол поворота молекул, p – давление, q – касательное напряжение, μ_1 и μ_2 – моментные напряжения, T – температура, ρ – плотность, j – момент инерции, κ – модуль объемного сжатия, α и γ – модули упругого сопротивления вращению и изменению кривизны, η – коэффициент вязкости, β – коэффициент теплового расширения, c – удельная теплоемкость, H – интенсивность источников тепла, $æ_{11}$, $æ_{12}$ и $æ_{22}$ – компоненты тензора теплопроводности: $æ_{11} = æ_1 \cos2\varphi + æ_2 \sin2\varphi$, $æ_{12} = (æ_1 - æ_2)\sin\varphi\cos\varphi$, $æ_{22} = æ_1 \sin2\varphi + æ_2 \cos2\varphi$, $æ_1$ и $æ_2$ – коэффициенты теплопроводности в направлении ориентации молекул и в поперечном направлении. Правые части f_1, f_2 и m обусловлены воздействием электрического поля. В систему (1) входят уравнения поступательного и вращательного движения, уравнение для угла поворота, уравнения состояния для давления и касательного напряжения, уравнения для моментных напряжений и уравнение анизотропной теплопроводности с переменными коэффициентами.

Численное решение системы уравнений (1) осуществляется с помощью метода двуциклического расщепления по пространственным переменным. На каждом временном интервале расщепление включает в себя 5 этапов. На 1-м и 5-м этапах (на разных полуинтервалах по времени) решается одномерная подсистема уравнений акустики и связанное уравнение теплопроводности в направлении x:

$$
\begin{gathered}
\rho v_{1,t} = -p_{,x} + f_1, \quad p_{,t} = -\kappa v_{1,x} + \beta T_{,t}, \quad q_{,t} = \alpha v_{2,x}, \quad \rho v_{2,t} = q_{,x}, \\
j\,\omega_{,t} = \mu_{1,x} + \bar{m}, \quad \mu_{1,t} = \gamma\,\omega_{,x}, \quad cT_{,t} = h_{1,x} - \beta T v_{1,x}, \quad h_1 = æ_{11}\,T_{,x} + æ_{12}\,T_{,y}.
\end{gathered}
$$

На 2-м и 4-м этапах расщепления решаются аналогичные уравнения в направлении y:

$$\rho v_{1,t} = -q_{,y}, \quad q_{,t} = -\alpha v_{1,y}, \quad \rho v_{2,t} = -p_{,y} + f_2, \quad p_{,t} = -\kappa v_{2,y} + \beta T_{,t},$$

$$j\omega_{,t} = \mu_{2,y} + \bar{m}, \quad \mu_{2,t} = \gamma \omega_{,y}, \quad c T_{,t} = h_{2,y} - \beta T v_{2,y}, \quad h_2 = æ_{12} T_{,x} + æ_{22} T_{,y}.$$

На 3-м этапе решается система уравнений по схеме Кранка–Николсон:

$$j\omega_{,t} = 2q, \quad \varphi_{,t} = \omega, \quad q_{,t} = -2\alpha(\omega + q/\eta), \quad c T_{,t} = 2q^2/\eta + H.$$

Такая конструкция метода расщепления обеспечивает устойчивость численного решения при выполнении одномерного условия Куранта. На этапах расщепления одномерные задачи решаются с помощью конечно-разностной схемы типа “предиктор–корректор”. Сначала реализуется шаг “предиктор” для подсистемы уравнений акустики, затем решается связанное уравнение теплопроводности и в заключение выполняется шаг “корректор” матричной системы, правая часть которой зависит от температуры и воздействий электрического поля. При решении акустических уравнений применяется разностная схема Годунова [3], а при решении уравнения теплопроводности – схема Иванова [4] с контролируемой диссипацией энергии. Температура в каждом направлении вычисляется с помощью трехточечной прогонки. Численный алгоритм без учета моментных напряжений и вязкости подробно описан в [5, 6].

Из системы (1) можно вывести подсистему уравнений второго порядка для касательного напряжения и угловой скорости:

$$q_{,tt} + \frac{2\alpha}{\eta} q_{,t} + 2\alpha\omega_{,t} = \frac{\alpha}{\rho}\Delta q, \quad \omega_{,tt} - \frac{2}{j} q_{,t} = \frac{\gamma}{j}\Delta\omega. \qquad (2)$$

Начальные данные для этой подсистемы таковы:

$$q\big|_{t=0} = q^0, \quad q_{,t}\big|_{t=0} = -2\alpha\left(\omega^0 + \frac{q^0}{\eta}\right), \quad \omega\big|_{t=0} = \omega^0, \quad \omega_{,t}\big|_{t=0} = \frac{2q^0}{j}, \qquad (3)$$

где ω_0, q_0 – константы, заданные в начальный момент времени.

Разработан параллельный вычислительный алгоритм для решения уравнений (2) с начальными данными (3). Граничные условия, моделирующие внешнее воздействие, задаются в терминах q и ω, а условия симметрии напряженного состояния жидкого кристалла – в терминах производных по x и y. Применяется конечно-разностная схема «крест» второго порядка точности, для которой доказано условие устойчивости. Проведено сравнение численного решения, полученного с помощью параллельной программы, и точного решения для одномерной задачи о действии касательного напряжения на границе области. Построены дис-

персионные кривые, отражающие зависимость фазовой скорости от частоты и зависимость декремента затухания от частоты [7, 8].

Воздействие электрического поля

Рассмотрим протяженный в горизонтальном направлении слой жидкого кристалла толщины h под действием периодического электрического поля, создаваемого зарядами на пластинах-обкладках конденсатора. Во внешней части слоя потенциал ϕ электрического поля $E = -\nabla\phi$ удовлетворяет уравнению Лапласа $\Delta\phi = 0$ при $x_2 < 0$ и при $x_2 > h$. Внутри слоя выполняется уравнение, учитывающее анизотропию среды: $\nabla \cdot (\varepsilon \cdot \nabla\phi) = 0$, где ε – тензор диэлектрической проницаемости жидкого кристалла, зависящий от ориентации молекул.

При наличии электрического поля молекулы жидкого кристалла испытывают действие объемных сил $f = (P \cdot \nabla) E$ и моментов $m = P \times E$, где $P = \varepsilon_0 \chi E$ – вектор электрической поляризации, $\chi = \varepsilon - I$ – тензор диэлектрической восприимчивости. Силы и моменты используются в качестве правых частей при решении системы уравнений акустического приближения жидкого кристалла (1).

Решение уравнения Лапласа во внешности слоя строится по методу прямых. Уравнение внутри слоя решается итерационным методом с помощью рекуррентного пересчета по формуле:

$$\varepsilon^0 \Delta\phi^* = \varepsilon^0 \Delta\phi - \nabla \cdot (\varepsilon \cdot \nabla\phi).$$

Анизотропная среда на каждом шаге заменяется однородной и изотропной средой с диэлектрической проницаемостью ε_0, с распределенными внутри нее зарядами. Правая часть вычисляется на предыдущем приближении электрического потенциала, величина ε_0 выбирается из соображений наискорейшей сходимости итераций. Условие окончания итерационного процесса $\|\phi^* - \phi\| \le \delta \|\phi\|$ с заданной погрешностью δ. Решение соответствующего дискретного уравнения Лапласа относительно нового приближения потенциала ϕ^* строится с помощью разложения Фурье. При нахождении коэффициентов разложения применяется метод трехточечной прогонки. Коэффициенты, через которые электрический потенциал определяется во внешности слоя, находятся из граничных условий на обкладках конденсатора и условий склейки решений на свободной границе жидкого кристалла. В итоге получается замкнутая система уравнений, которая решается методом LU-разложения.

Программная реализация вычислительного алгоритма

Ранее были разработаны параллельные программы для численного решения системы уравнений (1) без учета моментных взаимодействий [3, 4], а также для решения системы (2) [7, 8]. В настоящее время ведется отладка программы, позволяющей учесть влияние электрического поля на жидкий кристалл. В дальнейшем планируется объединить имеющиеся программные модули в единый комплекс программ для описания поведения жидких кристаллов с учетом термомеханических и электростатических внешних воздействий.

Программы реализуются на языке Си с применением технологии CUDA, позволяющей существенно увеличить вычислительную производительность благодаря использованию видеокарт. Для решения задач CUDA использует большое количество параллельных нитей, каждой из них соответствует один элемент вычисляемых данных. Подробное описание процедур и функций CUDA можно найти в [9, 10].

Программа для решения системы (1) имеет следующую структуру. Сначала на CPU задаются размерности конечно-разностной сетки и все необходимые константы, описываются одномерные массивы для 12 основных величин – компонент вектора скорости v_1 и v_2, угловой скорости ω, угла поворота φ, давления p, касательного напряжения q, моментных напряжений μ_1 и μ_2, температуры T, объемных сил f_1, f_2 и момента m, задаются начальные данные для этих величин. Одновременно на GPU выделяется память под массивы для искомых величин и всех необходимых вспомогательных величин. Затем необходимые константы и массивы копируются с CPU на GPU. В основном расчетном цикле по времени сначала выполняются процедуры, реализующие метод прямых для вычисления объемных сил и момента, обусловленных влиянием электрического поля, с применением дискретного преобразования Фурье, трехточечной прогонки и LU-разложения для решения СЛАУ. Далее на каждом шаге по времени и по каждому направлению запускаются ядра, последовательно реализующие двуциклическое расщепление: «предиктор» для системы уравнений акустики, граничные условия для уравнений акустики, вычисление прогоночных коэффициентов, прогонка для нахождения температуры, граничные условия, «предиктор» и «корректор» для потока. На 3-м этапе запускается ядро, реализующее схему Кранка–Николсон. После выполнения каждого из ядер производится барьерная синхронизация, чтобы обеспечить завершение вычислений каждой нитью до начала

выполнения следующего ядра. Результаты счета в контрольных точках копируются из GPU в CPU и записываются в файлы. В конце программы происходит освобождение памяти на GPU и CPU.

Структура параллельной программы для подсистемы (2) практически аналогична структуре программы для системы (1). Изменяется количество описываемых одномерных массивов и в основном расчетном цикле по времени производится только пересчет граничных условий в направлениях x и y и реализуется конечно-разностная схема “крест”.

Результаты расчетов

Для демонстрации работы программ проведена серия методических расчетов на высокопроизводительном вычислительном сервере Flagman с графическими вычислителями Tesla C2050 Института вычислительного моделирования СО РАН (общее число потоковых ядер CUDA на сервере – 3584).

На рис. 1 представлены результаты расчетов для задачи о действии трех Λ-образных импульсов давления на части левой границы расчетной области в виде квадрата. Локализованная нагрузка приложена в центре левой границы, остальные части границы области считаются свободными от напряжений. Нагрузка действует некоторое время, а потом граница полностью освобождается от напряжений. Начальные данные для всех величин нулевые. Разностная сетка – 2048×2048 ячеек. Расчеты проведены для жидкого кристалла 5ЦБ.

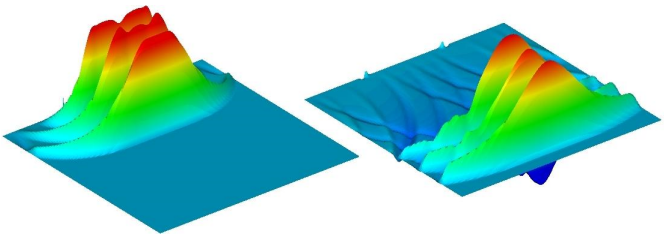


Рис. 1. Три Λ-образных импульса давления на части левой границы: линии уровня давления p (400-й и 1000-й шаги по времени)

На рис. 2 приведены результаты численного решения задачи о действии сосредоточенного касательного напряжения на верхней и нижней границах расчетной области. На левой и правой границах заданы условия

периодичности. В начальный момент времени касательное напряжение q и угловая скорость ω равны нулю. Вычисления проведены на прямоугольной области 10 × 5 мкм. Сверху нагрузка приложена в точке x_c = 7.5 мкм, снизу – x_c = 2.5 мкм. Размерность конечно-разностной сетки – 2560 × 2048 ячеек.

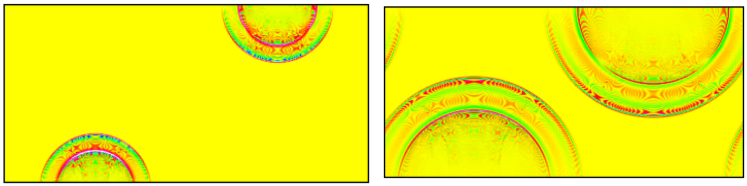


Рис. 2. Действие сосредоточенного касательного напряжения:
линии уровня угловой скорости ω (1000-й и 2000-й шаги по времени)

Чтобы оценить эффективность работы параллельных программ, проведено большое количество расчетов при разных размерностях сетки. Время счета параллельной программы сравнивалось со временем счета соответствующей последовательной программы. Для обеих программ зафиксировано ускорение примерно в 25 раз по сравнению с последовательной версией.

Литература

1. Blinov L.M. Structure and Properties of Liquid Crystals. Heidelberg ; New York ; Dordrecht ; London : Springer, 2011. 439 p. DOI 10.1007/978-90-481-8829-1.
2. Садовский В.М., Садовская О.В. Об акустическом приближении термомеханической модели жидкого кристалла // Физическая мезомеханика. 2013. Т. 16, № 3. С. 55–62.
3. Sadovskaya O.V. Numerical simulation of the dynamics of a liquid crystal in the case of plane strain using GPUs // AIP Conf. Proc. 2014. V. 1629. P. 303–310. DOI 10.1063/1.4902286.
4. Смолехо И.В. Параллельная реализация алгоритма для описания термоупругих волн в жидких кристаллах // Молодой ученый. 2015. № 11 (91). С. 107–112.
5. Годунов С.К., Забродин А.В., Иванов М.Я., Крайко А.Н., Прокопов Г.П. Численное решение многомерных задач газовой динамики. М. : Наука, 1976. 400 с.
6. Иванов Г.В., Волчков Ю.М., Богульский И.О., Анисимов С.А., Кургузов В.Д. Численное решение динамических задач упругопластического деформирования твердых тел. Новосибирск : Сиб. унив. изд-во, 2002. 352 с.

7. Смолехо И.В., Садовская О.В., Садовский В.М. Численное моделирование акустических волн в жидком кристалле с использованием технологии CUDA // Вычислительные технологии. 2017. Т. 22, спец. вып. 1. С. 87–98.
8. Smolekho I., Sadovskaya O., Sadovskii V. Numerical analysis of acoustic waves in a liquid crystal taking into account couple-stress interaction // CEUR Workshop Proc.: Mathematical and Information Technologies (MIT-2016). 2017. V. 1839. P. 473–486.
9. Боресков А.В., Харламов А.А. Основы работы с технологией CUDA. М. : ДМК Пресс, 2010. 232 с. ISBN 978-5-94074-578-5.
10. Farber R. CUDA Application Design and Development. 1st ed. Amsterdam ; Boston ; Heidelberg ; London ; New York ; Oxford ; Paris ; San Diego ; San Francisco ; Singapore ; Sydney ; Tokyo : Morgan Kaufmann / Elsevier, 2011. 332 p. ISBN 978-0-12-388426-8.

DOI: 10.17223/9785946216531/3

Параллельная реализация микромасштабной математической модели движения воздушных масс и переноса примеси*

Е.А. Данилкин, Л.Е. Грудович, Д.В. Лещинский

Томский государственный университет, Томск
ugin@math.tsu.ru

За последние два десятилетия произошел большой скачок в развитии вычислительной техники, на смену постоянному увеличению тактовой частоты процессора вернулись к идеям параллельной и конвейерной обработки данных. Появились доступные многоядерные процессоры и многопроцессорные компьютеры. И естественно возникает вопрос, а хорошо ли используемые на практике численные методы переносятся на новые вычислительные машины, заложен ли в них параллелизм? В работе рассмотрена параллельная реализация микромасштабной математической модели движения воздушных масс и переноса примеси в уличных каньонах. Параллельная реализация выполнена для системы с распределенной памятью и опирается на принцип геометрической декомпозиции сеточной области и использование библиотеки MPI при организации обмена сообщениями между параллельными процессами.

Ключевые слова: моделирование турбулентности, уличный каньон, перенос примеси.

Население планеты в основном проживает на урбанизированных территориях и процесс урбанизации продолжается. Это приводит к загрязнению атмосферного воздуха над городами и воздействует на окружающую среду и здоровье человека. Среди всех антропогенных источников загрязнения атмосферного воздуха наиболее значительную долю загрязнения в пределах городских территорий вносит автотранспорт.

В связи с постоянным ростом количества автотранспорта увеличивается и уровень загрязнения приземного слоя атмосферы выхлопными газами. Для понимания процессов переноса выбросов автотранспорта, происходящих внутри городских кварталов, наряду с приборным контро-

* Работа выполнена при финансовой поддержке гранта Президента РФ № МК-1723.2017.5 и при поддержке РФФИ и Администрации Томской области в рамках научного проекта № 16-41-700178 р_а.

лем состава атмосферного воздуха необходимо развитие систем мониторинга состояния окружающей среды, основанных на методах математического моделирования. Такие системы позволят предсказывать детальную структуру турбулентного течения воздушных масс и концентрации газовых составляющих примеси для различных метеоусловий и конфигураций городской застройки.

В настоящее время моделирование турбулентных течений в окружающей среде осуществляется, в основном, с использованием осредненных по Рейнольдсу уравнений Навье-Стокса (Reynolds Averaged Navier-Stokes) и переноса скалярных величин, для которых требуется решить проблему замыкания путем привлечения полуэмпирических моделей различного уровня сложности. Этот подход является значительно менее трудоемким с вычислительной точки зрения в сравнении с набирающими популярности LES и DES подходами [1–3], однако также нуждается в высокопроизводительных вычислительных алгоритмах.

Таким образом, целью данной работы является разработка и верификация двухмерной математической модели турбулентного течения и переноса примеси и численного метода ее решения адаптированного на использование многопроцессорной вычислительной техники. Основное назначение разрабатываемой модели – выполнение оперативных прогнозов и проведение параметрических расчетов.

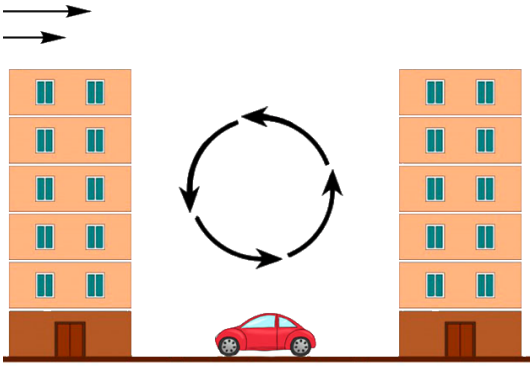


Рис. 1. Иллюстрация физической постановки задачи

Физико-математическую постановку задачи можно описать следующим образом. Рассматривается двухмерное нестационарное турбулент-

ное движение в приземном слое воздуха над неоднородной подстилающей поверхностью с элементами крупномасштабной шероховатости. Элементы шероховатости представляют собой неподвижные непроницаемые для потока прямоугольные препятствия (здания) (рис. 1). В области исследование допускается расположение точечных и линейных источников примеси постоянной интенсивности.

Математическая модель включает в себя осреднённые по Рейнольдсу уравнение неразрывности и уравнения Навье-Стокса [1]:

$$\frac{\partial\langle u_i\rangle}{\partial x_i} = 0; \tag{1}$$

$$\frac{\partial\langle u_i\rangle}{\partial t} + \frac{\partial\langle u_i\rangle\langle u_j\rangle}{\partial x_j} = \frac{1}{\rho}\frac{\partial\langle \mathrm{p}\rangle}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\vartheta\frac{\partial\langle u_i\rangle}{\partial x_j}\right) - \frac{\partial}{\partial x_j}\langle u_i' u_j'\rangle; \tag{2}$$

Здесь $\langle u_i\rangle$ – осредненные проекции вектора скорости на оси координат, $\langle \mathrm{p}\rangle$ – давление, ρ – плотность, ϑ – кинематическая вязкость воздуха, $\langle u_i' u_j'\rangle$ – тензор напряжений Рейнольдса, $i, j = 1{,}2$. Ось Ox_2 направлена вертикально вверх. Замыкание системы уравнений (1)–(2) проводится с использованием двухпараметрической «*k*-*ε*» модели и градиентно-диффузионной гипотезы Буссинеска [1]. Поле концентрации загрязняющих веществ определяется из решения уравнения переноса примеси, которое имеет вид:

$$\frac{\partial\langle \mathrm{c}\rangle}{\partial t} + \frac{\partial\langle u_j\rangle\langle \mathrm{c}\rangle}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\left(\frac{\vartheta}{Sc} + \frac{\vartheta_T}{Sc_T}\right)\frac{\partial\langle \mathrm{c}\rangle}{\partial x_j}\right) + \mathrm{S}; \tag{3}$$

где $\langle \mathrm{c}\rangle$ - осредненное значение концентрации примеси, S – функция, описывающая распределение точечных и линейных источников примеси, Sc – число Шмидта, ϑ_T – турбулентная вязкость (определяется из «*k*-*ε*» модели), Sc_T – турбулентное число Шмидта.

При расчёте течений вокруг зданий использовался метод фиктивных областей, суть которого заключается в том, что значения векторных и скалярных величин в области преграды равны нулю, и на границах фиктивных конечных объемов нет потоков диффузии [1]. Для задания значений турбулентных параметров вблизи подстилающей поверхности и поверхности элементов шероховатости используется метод пристеночных функций [1].

Дискретизация дифференциальной задачи осуществляется методом конечного объема с использованием разнесенной сетки [4]. При построении конечно объемной сетки используется алгоритм, в рамках которого

вся область исследования делится на подобласти и для каждой области задается свой признак: либо это несжимаемая среда (1), либо непроницаемое препятствие (9). При этом координаты границ подобластей подбираются так, чтобы разграничить (выделить) в отдельные подобласти непроницаемые участи внутри области исследования. На рис. 2 изображен пример области исследования, представляющий собой двумерную модель уличного каньона, где нижний правый и нижний левый квадраты представляют собой здания.

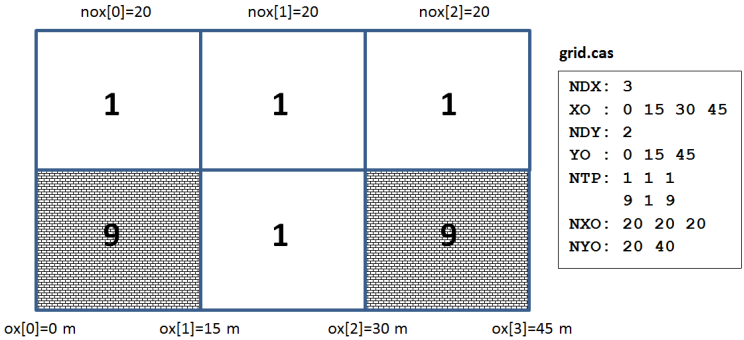


Рис. 2. Разделение области исследования на подобласти и пример содержания файла grid.cas для рассматриваемой области

Построение общей конечно объемной сетки осуществляется автоматически по данным из внешнего файла, который имеет следующую структуру. В первой строке указано общее число подобластей «NDX» в направлении оси OX (рис. 2). В следующей строке указаны «NDX+1» действительных чисел, являющихся координатами начала подобластей в направлении оси OX. Следующие две строки задают число подобластей в направлении оси OY и их координаты. Такие образом, область исследования покрывается набором смежных прямоугольников (подобластей). Далее в файле располагается матрица размера NDX* NDY, в которой заданы признаки подобластей: 1 – несжимаемая среда; 9 – непроницаемое препятствие. И в заключение следуют два массива целых чисел длиной NDX и NDY, в которых содержится число узлов в каждой подобласти в направлении осей OX, OY соответственно. На основании считанных данных строится единая глобальная сетка (сеточные линии), допускающая сгущения в выделенных подобластях, представляющих

наибольший интерес для исследования или требующих высокого разрешения с точки зрения моделируемых процессов.

После того как конечно объемная сетка построена, каждое дифференциальное уравнение интегрируется по каждому конечному объему. При дискретизации уравнений движения для компонент скорости и уравнений переноса использовалась явная аппроксимация по времени и противопотоковая схема для конвективных слагаемых. Таки образом, для уравнений движения и переноса получены явные разностных формулы, которые хорошо поддаются распараллеливанию [1].

При этом уравнения движения можно решать только в том случае, если поле давления задано или каким-либо образом найдено. Если при решении использовалось неверное поле давления, найденное поле скорости не будет удовлетворять уравнению неразрывности. Для согласования полей скорости и давления в гидродинамической части модели используется схема предиктор-корректор, в соответствии с которой явная схема для уравнения движения выполняет функцию предиктора, а коррекция поля скорости выполняется на основе решения уравнения Пуассона для поправки давления методом Зейделя с красно-черным упорядочиванием [1].

Тестирование вычислительного алгоритма проведено на задаче исследования течения в начальном участке плоского канала при низких числах Рейнольдса. Рассматривался ламинарный поток вязкой несжимаемой жидкости между двумя пластинами. Профиль скорости в начальном участке плоского канала сравнивался с аналитическим решением, полученным С.М. Таргом [5]. На рис. 3 представлено сравнение аналитического и численного решения, а именно сравнение осевой скорости в различных сечениях канала. Получен хороший уровень соответствия результатов численного моделирования и аналитического решения.



Рис. 3. Распределение осевой скорости при аналитическом и численном решении в различных сечениях канала

В качестве основного подхода распараллеливания выбрана геометрическая декомпозиция сеточной области. В рассматриваемом случае возможны два различных способа разделения значений сеточной функции по вычислительным узлам – одномерная или ленточная схема, двухмерное или блочное разбиение узлов вычислительной сетки.

На этапе декомпозиции, когда производится разделение данных на блоки для построения параллельного алгоритма, каждому процессорному элементу вместе с выделенной сеточной подобластью распределялись все значения сеточной функции, принадлежащие этой подобласти. На этапе планирования коммуникаций, когда устанавливаются связи между блоками, расчеты в которых выполняются параллельно, из-за используемого шаблона явной разностной схемы для вычисления очередного приближения в приграничных узлах каждой подобласти потребовались значения сеточной функции с соседнего граничащего процессорного элемента. Для этого на каждом процессорном элементе создавались фиктивные ячейки для хранения данных с соседнего процессорного элемента и организовывались пересылки этих граничных значений, необходимых для обеспечения однородности вычислений [1] (рис. 4). Для организации обмена сообщениями использовалась библиотека MPI.

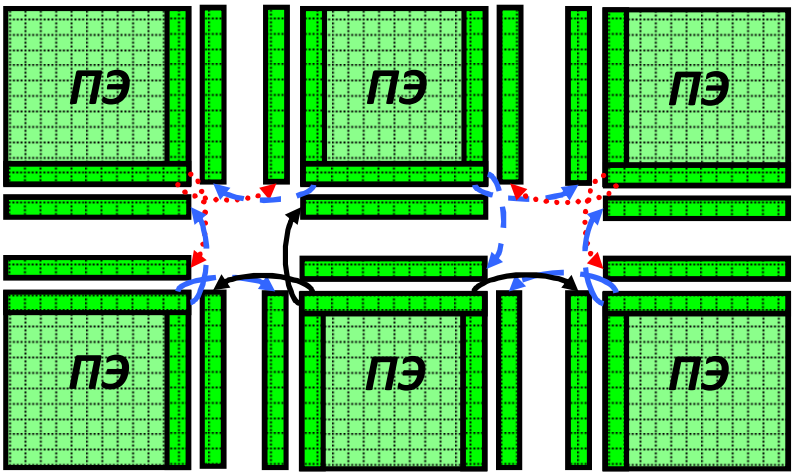


Рис. 4. Иллюстрация организации пересылок граничных сеточных значений при 2D-декомпозиции для обеспечения однородности вычислительного процесса

Время, затрачиваемое на пересылку данных при организации обменов, можно уменьшить, используя технологию опережающей рассылки, когда вычисления во внутренней области проходят на фоне пересылки уже вычисленных граничных значений сеточной функции. Использование многопроцессорной вычислительной техники позволило значительно сократить время вычислений при решении описанной задачи, с нескольких часов до нескольких минут, в зависимости от размера исследуемой области. В таблице представлено время, затрачиваемое на моделирование ламинарного течения в начальном участке плоского канали при использовании 1, 4, 16, 25, 64 и 100 процессов.

Время счета (2D-декомпозиция) для случая моделирования ламинарного течения в канале на сетке 120×40

Число процессов	1	4	16	25	64
Время счета, мин	38,72	23,58	2,67	1,18	0,85

Заключение

Для исследования турбулентных течений в уличных каньонах разработана и верифицирована двумерная математическая модель турбулентного течения и численный метод ее решения. Верификация проведена на задаче моделирования ламинарного течения в начальном участке плоского канала. Разработан параллельный алгоритм решения рассматриваемой задачи.

Литература

1. Старченко А.В., Нутерман Р.Б., Данилкин Е.А. Численное моделирование турбулентных течений и переноса примеси в уличных каньонах. Томск : Изд-во Томского государственного университета, 2015. 252 с.
2. Sagaut P. Large eddy simulation for Incompressible Flow 3rd ed. An Series: Scientific Computation, 2006. 556 p.
3. Danilkin E.A., Starchenko A.V. Large eddy simulation of turbulent flow and of pollutant transport in a street canyon // Proc. SPIE 9680, 21th International Symposium on Atmospheric and Ocean Optics: Atmospheric Physics. 2015.
4. Патанкар С. Численные методы решения задач теплообмена и динамики жидкости / пер. с англ. М. : Энергоатомиздат, 1984. 149 с.
5. Жукаускас А.А. Конвективный перенос в теплоприемниках. М. : Наука, 1982. 472 с.

DOI: 10.17223/9785946216531/4

Использование векторизованного формата хранения разреженных матриц для увеличения скорости матрично-векторного умножения*

В.С. Гладких[1], С.Г. Пудов[2]

[1] Институт вычислительной математики и математической геофизики СО РАН, Новосибирск
[2] Сибирский государственный университет телекоммуникаций и информатики, Новосибирск
gladvs_ru@mail.ru

Во многих задачах математического моделирования самой затратной частью являются решения систем линейных алгебраических уравнений (СЛАУ) с разреженной матрицей. Для крыловских итерационных методов решения СЛАУ ключевым фактором скорости является производительность матрично-векторного умножения (SpMV). В работе представлена модификация популярного CSR формата хранения разреженной матрицы, учитывающая особенности современных векторных архитектур и подсистем памяти, позволяющая достигать заметного ускорения матрично-векторного умножения. Выделен подкласс разреженных матриц, для которых разумно использовать представленный формат. Формат и SpMV реализованы как расширение библиотеки KRYLOV.
Ключевые слова: матрично-векторное умножение, формат хранения разреженной матрицы, векторизация.

Скорость и производительность (performance) решения систем линейных алгебраических уравнений (СЛАУ) является одной из ключевых составляющих многих современных программных пакетов. Используемые матрицы, возникающие при моделировании разнообразных физических процессов, являются сильно разреженными, то есть количество ненулевых элементов в строке матрице A невелико и обычно не превышает 50. Для эффективного хранения разреженных матриц используются различные форматы, такие как: разреженный строчный (compressed sparse row, CSR) и координатный (COO) для произвольных матриц, а также разреженный блочный (block compressed sparse row, BSR), диагональный (DIA)

* Работа поддержана грантами РНФ № 14-11-00485П и РФФИ № 16-29-1512 офи_м.

и EllPack (ELL) для матриц специальной структуры [2]. Стоит отметить, что для крыловских итерационных методов решения СЛАУ наиболее затратной операцией являются умножение разреженной матрицы (SpMV) на вектор, которая в итоге и определяет производительность и скорость решения СЛАУ. Поэтому уделяется повышенное внимание вопросам оптимизации SpMV на современных вычислительных системах [1, 3, 4].

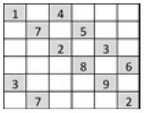
Стремительный рост вычислительных мощностей, появление новых микроархитектур и способов параллелизации привело к тому, что для большинство форматов хранения разреженных матриц невозможно использовать все возможности современных вычислительных систем, в том числе и векторизацию. При этом скалярный код не обеспечивает необходимой производительности. Поэтому для оптимизации вычислений постоянно разрабатываются или модифицируются существующие форматы хранения разреженных матриц [5, 6]. Целью данной работы было продемонстрировать преимущества использования формата хранения данных ориентированного на современные особенности микроархитектуры процессоров и подсистемы памяти. Для чего рассматривался формат являющийся векторизованной версией CSR – vector compressed sparse row (VSR). Для матриц подходящей структуры он может существенно ускорить вычисления как за счет снижения числа выполняемых операций процессора, так и за счет уменьшения объема данных, необходимых для хранения матрицы.

Для начала рассмотрим такой популярный формат хранения матриц как CSR: хранятся только ненулевые элементы матрицы по строкам. Пусть N – число строк матрицы A, а NNZ – число ненулевых элементов. Тогда для хранения значений элементов и их столбцов используются два массива: a[1:nnz] и ja[1:nnz], в которые элементы заносятся построчно, начиная от первой строчки и заканчивая последней, при этом индекс первого элемента каждой строки хранится в дополнительном специальном массиве ia[1:N+1]. В последнем элементе массива ia хранится общее количество ненулевых элементов в матрице. Таким образом, для матрицы A можно написать псевдокод умножения матрицы на вектор $y = A \cdot x$ следующим образом:

Цикл по всем i от 0 до N-1 включительно
y[i] = 0
Цикл по всем j от ia[i] до ia[i+1]-1 включительно
*y[i] = y[i] + a[j]*x[ja[j]].*

VSR формат аналогичен CSR формату, за тем исключением, что одним элементом является короткая диагональ фиксированного размера равного v. В массивы ja и ia помещаются только данные первого элемента. Для лучшей наглядности проиллюстрируем на псевдо-коде и примере, приведённом на рис. 1:

Цикл по всем i от 0 до N-1 включительно
y[i] = 0;
Цикл по всем i от 0 до N-1 включительно
Цикл по всем j от ia[i] до ia[i+1]-1 включительно
Цикл по k от 0 до v
*y[i+k] = y[i+k] + a[j+k]*x[ja[j]+k].*



CSR:
ia = (0,2,4,6,8,10,12),
ja =(0,2,1,3,2,4,3,5,0,4,1,5),
a = (1,4,7,5,2,3,8,6,3,9,7,2)

VSR (v=2):
ia = (0,2,2,4,4,6),
ja = (0,2,2,4,0,4),
a = (1,7,4,5,2,8,3,6,3,7,9,2)

Рис. 1. Пример разреженной матрицы с CSR и VSR представлением

Этот формат позволяет использовать такие особенности современных микропроцессоров как векторизация и загрузка/выгрузка из памяти строки кэша (cache line), при этом размер коротких диагоналей должен подбираться в зависимости от архитектуры процессора. Например, если ориентироваться только на векторизацию операций, то длина короткого вектора будет равна 4-м вещественным числам двойной точности для наборов инструкции AVX и AVX2 (256 бит), и 8 вещественным числам двойной точности для более поздних (512 бит). Доступ к данным определяется архитектурой кэша, строки которого длиной 512 бит содержат ровно 8 вещественных чисел двойной точности. Поэтому эффективное использование как векторизации, так и операции с памятью достигается уже при размере диагонали равным 8 (v = 8) в формате VSR. Пример, приведенный на рис. 2, показывает преимущества, которые дает новый формат по сравнению с CSR.

Использование VSR формата позволяет достигать потенциального ускорения до 8-ми раз для современных вычислительных устройств. В реальности полученное ускорение будет гораздо меньше, так как разнообразные механизмы, реализованные в процессоре и компиляторах программ, позволяют избегать полного штрафа за неиспользование векторизации. К таким механизмам, частично убирающим штраф за отсутствие

векторизации, в данном случае можно отнести многоуровневую иерархию памяти, компиляторную оптимизацию вычислительных операций, внеочередное исполнение и прочее. Тем не менее, использование векторизованного формата хранения данных должно приводить и дает выигрыш при вычислении умножения разреженной матрицы на вектор.

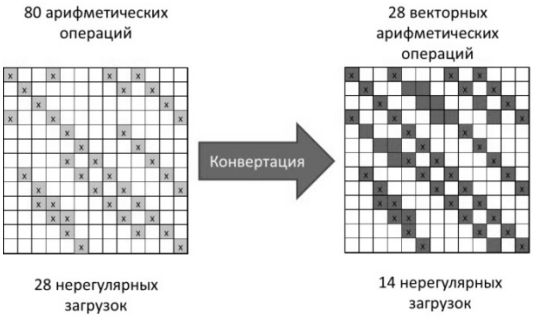


Рис. 2. Пример, демонстрирующий преимущество матрицы в VSR(v=4) формате против такой же матрицы в CSR формате в арифметических операциях и загрузках из памяти. Пустыми закрашенными клеточками обозначены хранящиеся в памяти дополнительные нулевые элементы

Следует обратить внимание что VSR в отличие от CSR не является подходящим форматом хранения для всех типов матриц, так как далеко не всегда матрица может быть эффективна представима в VSR формате. При выполнении конвертации из CSR в VSR необходимо добавлять нулевые элементы, если соответствующие элементы не хранятся в CSR формате. Эффективности представления матрицы в VSR формате определяется тем, какое дополнительное количество нулей матрицы необходимо хранить. Наиболее подходящими являются матрицы, портрет которых преимущественно состоит из диагоналей, возможно частичных. Для таких матриц характерно небольшое количество добавленных ненулевых элементов относительно общего числа элементов при переходе к VSR формату. В общем случае конвертация матриц в VSR формат может приводить к увеличению времени, наглядный пример такой матрицы приведен на рис. 3.

В качестве синтетического теста использовался генератор матрицы, соответствующей 27-точечному шаблону при аппроксимации в кубике с равномерной сеткой. В качестве реальных задач был выбран большой

набор матриц, загруженных с коллекции матриц FLORIDA [7], из которых были выбраны некоторые характерные матрицы: TSOPF/TSOPF_RS_b162_c3, TSOPF/TSOPF_FS_b162_c1, Schenk_IBMNA/c-64b, AMD/G2_circuit, GHS_indef/olesnik0, IPSO/HTC_336_4438 и IPSO/HTC_336_9129, демонстрирующие как недостатки, так и достоинства представленного VSR формата.

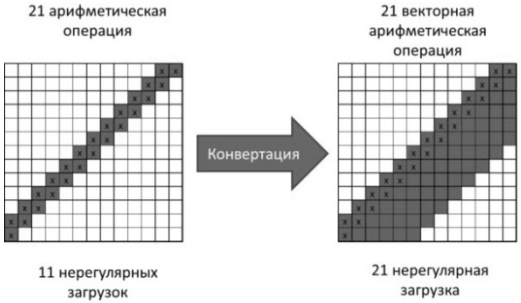


Рис. 3. Пример, демонстрирующий недостатки VSR формата для ряда матриц. Пустыми закрашенными клетками обозначены хранящиеся в памяти дополнительные нулевые элементы

Как видно из представленного замера производительности, предложенный формат хранения матрицы хорошо подходит для использования в случае синтетических матриц (рис. 4).

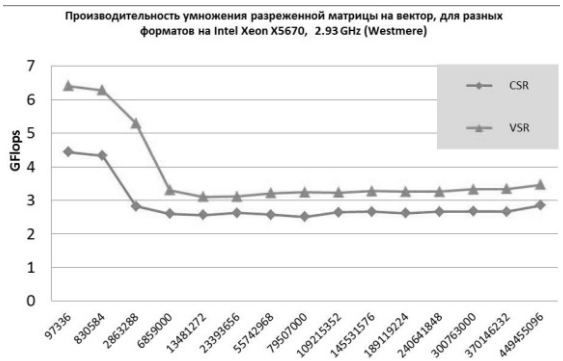


Рис. 4. Производительность CSR и VSR формата для 27-диагональной матрицы

Для ряда матриц, возникающих в реальных приложениях, использование VSR формата способно как существенно ускорить матрично-векторное умножение, так и замедлить его в случае применения к матрицам с неподходящей структурой, примеры приведены на рис. 5.

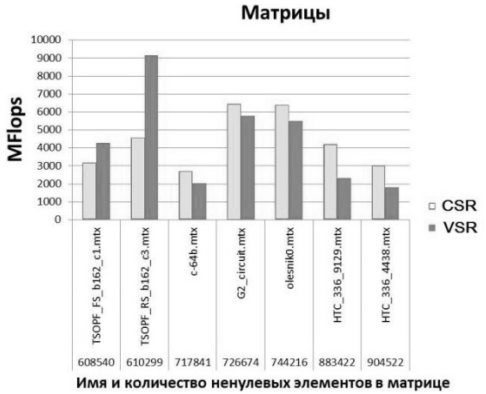


Рис. 5. Производительность CSR и VSR формата
для набора характерных матриц с Florida matrix collection

Предложенный VSR формат хранения разреженных матриц показал высокую эффективность на генерируемых матрицах и на части матриц из Florida matrix collection, имеющих портрет, представимый в виде набора диагоналей, возможно частичных, и с небольшим количеством элементов вне диагоналей. Данный класс матриц достаточно часто встречается при решении различных задач моделирования физических процессов и позволяет использовать VSR без значительного увеличения хранимых элементов матрицы. На современных вычислительных системах наблюдается ускорение от 20 до 200 и более процентов. VSR матрично-векторное умножение и конвертация из CSR в VSR были реализованы как расширение библиотеки KRYLOV[8].

Литература

1. Bell N., Garland M. Efficient Sparse Matrix-Vector Multiplication on CUDA 2008. С. 1–32.
2. Bell N., Garland M. Implementing sparse matrix-vector multiplication on throughput-oriented processors // Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis - SC ’09. 2009. № 1. С. 1.

3. Choi J.W., Singh A., Vuduc R.W. Model-driven autotuning of sparse matrix-vector multiply on GPUs // ACM SIGPLAN Notices. 2010. № 5 (45). P. 115.
4. Eberhardt R., Hoemmen M. Optimization of block sparse matrix-vector multiplication on shared-memory parallel architectures // Proceedings - 2016 IEEE 30th International Parallel and Distributed Processing Symposium, IPDPS 2016. 2016. P. 663–672.
5. Liu L. [и др.]. LSRB-CSR: A low overhead storage format for SpMV on the GPU systems // Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS. 2016. (2016–January). P. 733–741.
6. Maggioni M., Berger-Wolf T. Optimization techniques for sparse matrix-vector multiplication on GPUs // Journal of Parallel and Distributed Computing. 2016. (93–94). P. 66–86.
7. Florida matrix collections. [Электронный ресурс]. URL: https://sparse.tamu.edu/
8. Бутюгин Д.С., Гурьева Я.Л., Ильин В.П. [и др.]. Функциональность и технологии алгебраических решателей в библиотеке Krylov. // Вестник ЮУрГУ. Серия «Вычислительная математика и информатика». 2013. Т. 2. С. 92–105.

DOI: 10.17223/9785946216531/5

Вычисление сильно-составных чисел

В.М. Зюзьков

Томский государственный университет, Томск
vmz@math.tsu.ru

Положительное целое n называется сильно-составным числом, если для целых m < n выполнено τ(m) < τ(n), где τ(x) обозначает количество положительных делителей числа x. Используя результаты С. Рамануджана, был создан алгоритм для вычисления сильно-составных чисел. Вычисления проводились в системе Mathematica с помощью программирования на языке Wolfram. Исследование полученных первых 435 чисел привело к гипотезе о распределении сильно-составных чисел в целом. На основании гипотезы предложен эффективный алгоритм, с помощью которого вычислены первые 1003 сильно-составные числа. С помощью математических экспериментов найдена приближенная формула для оценки количества сильно-составных чисел меньших данного натурального числа.
Ключевые слова: сильно-составные числа, математические эксперименты, Mathematica.

Определение. Положительное целое n называется *сильно-составным числом*, если для всех целых $m < n$ выполнено $\tau(m) < \tau(n)$.

Понятие сильно-составного числа ввел Сриниваса Рамануджан [1].

Легко проверить, что первые шесть сильно-составных чисел есть 1, 2, 4, 6, 12, 24. Принято считать, что 1 и 2 являются сильно-составными, хотя они даже не составные. Современные знания о сильно-составных числах описаны в [2].

Условимся использовать сокращение «*hc*-число» вместо термина «сильно-составное число».

Следующие два утверждения доказываются легко.

Предложение 1. Пусть h – *hc*-число и для натурального числа m имеем $\tau(h) < \tau(m)$, тогда существует такое *hc*-число k, что $h < k \le m$.

Следствие. Сильно-составных чисел бесконечно много.

Пусть h_n обозначает n-е по порядку возрастания *hc*-число, т.е. $h_1 = 1$, $h_2 = 2$, $h_3 = 4$, $h_4 = 6$ и т.д. Можно указать верхнюю границу для *hc*-числа в зависимости от его порядкового номера.

Предложение 2. Пусть n – натуральное число. Тогда, $h_n \le 2^n$.

Эта оценка конечно очень завышенная. Следующая простая программа на языке Wolfram [3, 4] выдает все *hc*-числа, не превосходящие 10^8.

```
⇒ s = {}; a = 0;
  Do[b = DivisorSigma[0, n]; If[b > a, a = b; s = {s, n}], {n, 10^8}];
  Flatten[s]
```

{1, 2, 4, 6, 12, 24, 36, 48, 60, 120, 180, 240, 360, 720, 840, 1260, 1680, 2520, 5040, 7560, 10080, 15120, 20160, 25200, … , 4324320, 6486480, 7207200, 8648640, 10810800, 36756720, 43243200, 61261200, 73513440}

На месте многоточия пропущено 23 числа. Программа, реализуя алгоритм «грубой силы», перебирает последовательно натуральные числа, вычисляя каждый раз сумму делителей, и поэтому работает очень медленно.

Для создания более быстрого алгоритма рассмотрим некоторые свойства *hc*-чисел.

Пусть p_i, $i > 1$, есть строго возрастающая последовательность простых чисел. Факторизации числа

$$n = p_{i_1}^{\alpha_1} \cdots p_{i_r}^{\alpha_r}, \quad 1 \le i_1 < \cdots < i_r,$$

на простые множители сопоставим число

$$\boldsymbol{n} = p_1^{\beta_1} \cdots p_r^{\beta_r}, \quad \beta_1 \ge \beta_2 \ge \cdots \ge \beta_r, \tag{1}$$

где показатели степеней β_i получены из α_i переупорядочиванием в убывающем порядке. Например, для $n = 3^5 \cdot 7^2 \cdot 13^3 \cdot 17^2 \cdot 23^9$ получаем число $\boldsymbol{n} = 3^9 \cdot 7^5 \cdot 13^3 \cdot 17^2 \cdot 23^2$.

Теорема 1 (Рамануджан [1]). Имеем $\tau(\boldsymbol{n}) = \tau(n)$ и $\boldsymbol{n} \le n$.

Следствие. Любое *hc*-число имеет форму

$$n = 2^{\alpha_2} 3^{\alpha_3} \cdots p^{\alpha_p}, \quad \alpha_2 \ge \alpha_3 \ge \cdots \ge \alpha_p \ge 1. \tag{2}$$

С. Рамануджан оценил значения α_i в факторизации (2).

Теорема 2 (Рамануджан [1]). Пусть $h = 2^{\alpha_2} 3^{\alpha_3} \cdots p^{\alpha_p}$ есть *hc*-число и P – простое число, которое следует после p. Тогда для каждого $q \le p$ имеем

$$\left\lfloor \frac{\log p}{\log q} \right\rfloor \le \alpha_q \le 2 \left\lfloor \frac{\log P}{\log q} \right\rfloor \text{ и } \alpha_p = 1 \text{ для } p > 3. \tag{3}$$

Если p – наибольший простой делитель в разложении (2) *hc*-числа h, то теорема 2 позволяет оценить диапазоны значений показателей α_i. Была написана программа на языке *Wolfram*, которая для простого числа p выдает границы диапазонов возможных значений α_i.

Например, для $p = 7$ имеем $2 \le \alpha_2 \le 6$, $1 \le \alpha_3 \le 4$, $1 \le \alpha_5 \le 2$, и $\alpha_7 = 1$.

Для описания алгоритмов вычисления *hc*-чисел введем необходимую терминологию и обозначения.

Число n называется *потенциально сильно-составным числом*, если для n выполнены условия (2) и (3). Множество всех потенциально сильно-составных чисел будем обозначать Λ.

Если $n \in \Lambda$, то функция $maxpd(n)$ выдает в качестве значения наибольший простой делитель n.

Если p – простое число, то обозначим

$min(p)$ = минимум $\{n \mid n \in \Lambda$ и $maxpd(n) = p\}$,

$max(p)$ = максимум $\{n \mid n \in \Lambda$ и $maxpd(n) = p\}$.

Если p – простое число, то определим множество $\Lambda_p = \{n \mid n \in \Lambda$ и $maxpd(n) = p\}$. Очевидно, Λ_p – подмножество замкнутого интервала $[min(p), max(p)]$.

Множество чисел Λ_p вычисляется с помощью вложенного цикла. Если $p > 3$, то вложенность на 1 меньше порядкового номера простого числа p.

Из определения *hc*-чисел следует, что их вычислять можно только по очереди в порядке возрастания. Для этого достаточно просматривать в возрастающем порядке потенциально сильно-составных чисел. Рассматривая первые 56 *hc*-чисел, вычисленные с помощью «грубой силы», обнаруживаем, что если h_1 и h_2 – какие-то два подряд идущих *hc*-числа, то может быть выполнено любое из соотношений $maxpd(h_1) = maxpd(h_2)$, $maxpd(h_1) < maxpd(h_2)$ или $maxpd(h_1) > maxpd(h_2)$. Это свойство естественно сохраняется, если h_1 и h_2 – два подряд идущих потенциально сильно-составных числа.

Предлагаемые ниже алгоритмы вычисления *hc*-чисел представляют все множество Λ в виде объединения непересекающихся диапазонов потенциально сильно-составных чисел. Эти диапазоны могут быть открытыми, замкнутыми или полуоткрытыми интервалами. Таким образом, получение всех *hc*-чисел сводится к вычислению *hc*-чисел в последовательных интервалах.

Для определенности возьмем некоторое простое число q и рассмотрим, как разбивается множество $[2, min(q)) \cap \Lambda$ на интервалы. Для этого сначала определим числа, которые будут концами этих интервалов. Пусть последовательность B состоит из упорядоченных по возрастанию элементов множества

$\{b \mid \exists$ (простое $p < q$) $((b = min(p)) \vee (b = max(p)))$ & $(b < min(q))\} \cup \{min(q)\}$.

Заметим, что для каждого b значение $maxpd(b)$ равно некоторому простому числу, не превосходящему q. Однозначно для каждого элемента $b \in B$ определяется, в какой соседний интервал S_i входит b в качестве граничного элемента. Каждый интервал S_i будет характеризоваться своим множеством $D_i = \{maxpd(n) \mid n \in S_i\}$.

Алгоритм *A*. Вычисление *hc*-чисел из интервала разбиения множества $[2, min(q)) \cap \Lambda$. Входные данные: 1) некоторый интервал S с концами $b1$ и $b2$ (конец интервала может входить или не входить в интервал); 2) множество простых чисел $D = \{maxpd(n) \mid n \in S\}$; 3) наибольшее *hc*-число h, меньшее всех чисел из интервала S. Выход: Г – множество *hc*-чисел из интервала S.

Для каждого $p \in D$ выполняем следующее:

1*a*. Находим множество M всех последовательностей натуральных чисел $\alpha_2, \alpha_3, \alpha_5, ..., \alpha_p$, удовлетворяющих условиям (2) и (3).

1*b*. Для каждой последовательности $\alpha_2, \alpha_3, \alpha_5, ..., \alpha_p$, из M вычисляем число

$$n = 2^{\alpha_2} 3^{\alpha_3} \cdots p^{\alpha_p}.$$

Множество полученных таким образом чисел n есть множество Λ_p.

1*c*. Учитывая границы интервала S, получаем множество потенциально сильно-составных чисел $S \cap \Lambda_p$.

Объединяем все множества $S \cap \Lambda_p$ полученные на шаге 1:

$$\bigcup_{p \in D} (S \cap \Lambda_p) = S \cap \Lambda.$$

Упорядочиваем по возрастанию множество $S \cap \Lambda$. Получили возрастающую последовательность всех потенциально сильно-составных чисел из интервала S. Обозначим эту последовательность через T.

Из последовательности T извлекаем только *hc*-числа. Создаем последовательность Г по возрастанию *hc*-чисел. Первоначально, $\Gamma = \{h\}$. Последовательно проходим числа из T и каждый раз, когда встречаем число n со свойством $\tau(n) > \tau(m)$ (m есть последнее число в Г), добавляем n в конец Г. Завершая шаг 4, удаляем первый элемент h из Г.

Программа на языке *Wolfram* для алгоритма *A* выполнялась на сервере с двумя процессорами Intel Xeon E5-2620 v2. Параллельно вычислялись множества чисел Λ_p, для простых $p \le 83$. Диапазон чисел $[2, min(83)) \cap \Lambda$ был разбит на 38 интервалов. Хотя использовалось 64 Гбайт оперативной памяти, вычисления проходили очень медленно, по-видимому, из-за недостатка памяти. Были найдены первые 435 *hc*-чисел.

Для создания эффективного алгоритма было исследовано, какие значения принимает наибольший простой делитель для найденных *hc*-чисел. На рис. 1 изображен график зависимости наибольшего простого делителя сильно-составного числа h_n от его номера n.

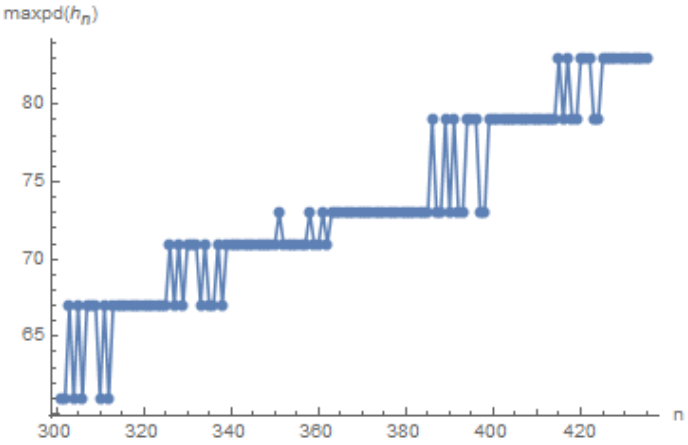


Рис. 1. Зависимость $maxpd(h_n)$ от n

Вид этой зависимости сохраняется для всех найденных 435 *hc*-чисел. Изучение графика приводит к гипотезе, но прежде введем обозначение

$$H(p) = \{n \mid n - hc\text{-число и } maxpd(n) = p\}.$$

Гипотеза. Пусть p_0, p_1 и p_2 – три подряд идущих простых числа. Обозначим n_1, n_2 – наименьшие числа множеств $H(p_1)$ и $H(p_2)$ соответственно. Тогда, если n – *hc*-число в промежутке $n_1 \le n < n_2$, то $maxpd(n)$ равно p_0 или p_1.

Это предположение сразу приводит к эффективному алгоритму.

Алгоритм *B*. Вход: p_0, p_1 и p_2 – три подряд идущих простых числа и n_1 – наименьшие число множества $H(p_1)$. Выход: n_2 и все *hc*-числа n в интервале $n_1 \le n \le n_2$.

Находим $\Lambda(p_0)$, $\Lambda(p_1)$ и $\Lambda(p_2)$.

Упорядочиваем по возрастанию множество потенциально сильно-составных чисел из объединения

$$T = \{n \mid n \in \Lambda(p_0) \text{ и } n > n_1\} \cup \Lambda(p_1) \cup \Lambda(p_2).$$

Находим последовательно все *hc*- числа из множества T пока не обнаружим первое сильно-составное число n, для которого $maxpd(n) = p_2$. Полагаем n_2 равным этому n.

Применив алгоритм B для входа p_0, p_1 и p_2 и *hc*-числа n_1 и получив результат, можно запустить этот алгоритм для входа p_1, p_2 и p_3 и *hc*-числа n_2, где p_3 – следующее простое число большее p_2. Алгоритм B не нуждается в параллельных вычислениях и не требует для выполнения много оперативной памяти. Фактически соответствующая программа на языке Wolfram выполнялась на домашнем компьютере.

Было вычислено 1003 сильно-составных чисел. Число h_{1003} равно

594632450474407886206387614697350771058387730170444871793178
02012440841792000.

Это число имеет 77 цифр, его факторизация есть $2^9\times3^5\times5^3\times7^3\times11^2\times$
$\times13^2\times17^2\times19\times23\times29\times31\times37\times41\times43\times47\times53\times59\times61\times67\times71\times73\times79\times83\times$
$\times89\times97\times101\times103\times107\times109\times113\times127\times131\times137\times139\times149\times151\times157\times$
$\times163\times167$ и количество делителей равно 111 325 552 312 320.

Так как *hc*-числа растут очень быстро, то более наглядным является график зависимости $\ln h_n$ от номера n (рис. 2).

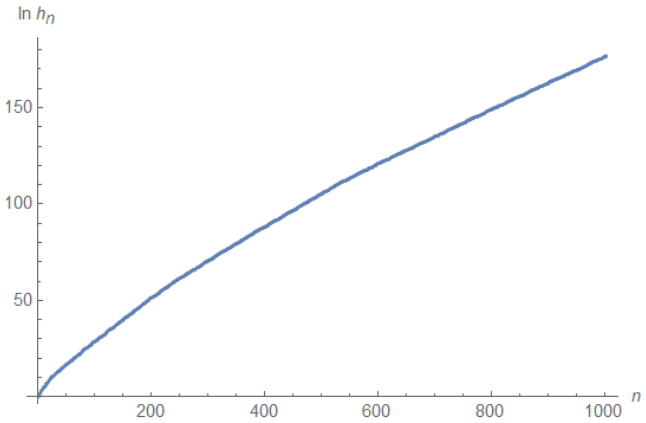


Рис. 2. Рост $\ln h_n$

Определим значение функции $\gamma(x)$ для натурального числа x как количество *hc*-чисел, не превосходящих x. Будем считать, что значения функции $\gamma(x)$ известны для всех $x \le h_{1003}$, но хотелось бы иметь приближенную

формулу, аппроксимирующую $\gamma(x)$. Проводя математические эксперименты на языке Wolfram, и используя метод наименьших квадратов, была найдена функция $\ln x \to 1.77457 \ln x + 0.293921(\ln x)^{3/2}$, аппроксимирующая функцию $\ln x \to \gamma(x)$. На рисунке 3 изображены вместе графики функций $\ln x \to \gamma(x)$ (толстая кривая) и $\ln x \to 1.77457 \ln x + 0.293921(\ln x)^{3/2}$ (тонкая кривая). На оси абсцисс располагаются значения $\ln x$. Эти графики визуально совпадают.

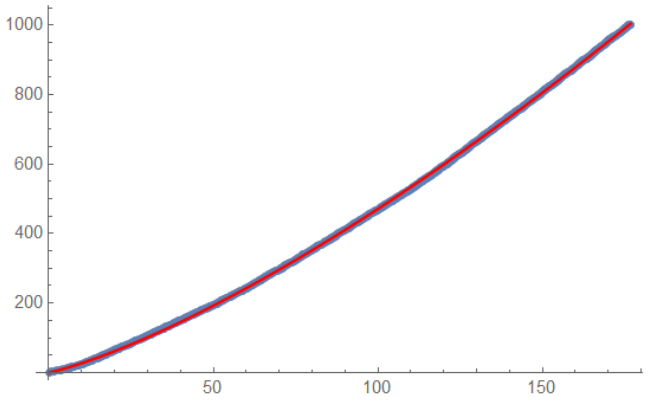


Рис. 3. Графики $\ln x \to \gamma(x)$ и $\ln x \to 1.77457 \ln x + 0.293921(\ln x)^{3/2}$

Таким образом, имеем приближенное равенство

$\gamma(x) \approx 1.77457 \ln x + 0.293921(\ln x)^{3/2}$.

Автор благодарит А. А. Барта за выполнение программ для алгоритма *A* на сервере ММФ.

Литература

1. Ramanujan S., Highly Composite Numbers, Proc. London Math. Soc. XIV (1915), pp. 347–409. URL: http://ramanujan.sirinudi.org/Volumes/published/ram15.pdf.
2. Wolfram MathWorld: Highly Composite Number. URL: http://mathworld.wolfram.com/HighlyCompositeNumber.html.
3. Зюзьков В. М. Начала компьютерной алгебры: учебное пособие. Томск: Издательский Дом Томского государственного университета, 2015, 128 с. URL: http://vital.lib.tsu.ru/vital/access/manager/Repository/vtls:000509029.
4. Hastings C., Mischo K., Morrison M. Hands-on Start to Wolfram Mathematica and Programming with the Wolfram Language. 2nd ed. Wolfram Media Inc., 2016. 484 p.

DOI: 10.17223/9785946216531/6

Распараллеливание явных и неявных разностных схем на примере численного решения уравнения теплопроводности*

Д.В. Лещинский, Е.А. Данилкин

Томский государственный университет
360flip182@gmail.com

Работа посвящена математическому моделированию процесса теплообмена между двумя телами с различными теплофизическими свойствами. Для решения данной задачи выбраны явная и неявная разностные схемы. Проведено исследование аппроксимации, устойчивости и сеточной сходимости полученных разностных схем. Проведено распараллеливание полученного алгоритма. Также проведен сравнительный анализ между численным решением, полученным с использованием разработанной программы, и численным решением, полученным при моделировании данного процесса в программе ANSYS Fluent.
Ключевые слова: уравнение теплопроводности, явная и неявная разностные схемы, параллельные вычисления, ANSYS Fluent.

Существует множество задач, решение которых невозможно получить, используя только лишь аналитический подход. Тогда для получения решения приходится использовать как аналитический подход, так и применять методы численного решения. Однако в ходе решения подобных задач возникает ряд проблем, связанных с объёмом обрабатываемой информацией, что, в свою очередь, влечет к значительному увеличению промежутка времени, затрачиваемого на решение поставленной задачи. Использование многопроцессорной вычислительной техники позволяет значительно быстрее обрабатывать огромные массивы информации в отличие от последовательных вычислительных систем [1, 2].

В данной работе будет представлено численное решение двумерной задачи теплопроводности, распараллеливание полученного алгоритма решения и верификация построенной модели.

* Работа выполнена при финансовой поддержке гранта Президента РФ № МК-1723.2017.5 и при поддержке РФФИ и Администрации Томской области в рамках научного проекта № 16-41-700178 р_а.

Физико-математическую постановку задачи можно описать следующим образом. Моделируется процесс распространения тепла при контактном взаимодействии стальной балки, нагретой до температуры 1500℃, с поверхностью земли, имеющей температуру 20℃ (рис. 1). С течением времени балка остывает за счет теплообмена с окружающей средой и поверхностью земли.

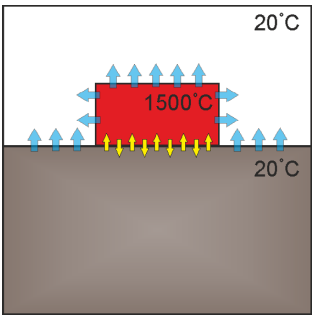


Рис. 1. Иллюстрация физической постановки задачи

Распространение тепла в теле описывается уравнением теплопроводности:

$$\frac{\partial T}{\partial t} = a\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right), \quad (1)$$

где a – коэффициент температуропроводности ($м^2/с$), представляющий собой физическую величину, характеризующую скорость изменения (выравнивания) температуры вещества в неравновесных тепловых процессах, который определяется по формуле [3]:

$$a = \frac{\lambda}{\rho C}, \quad (2)$$

где ρ – плотность (кг/ $м^3$), C – удельная теплоемкость (Дж/кг·К), λ – коэффициент теплопроводности (Вт/м·К). Так как в процессе теплообмена участвуют два тела, то для них будут использоваться разные уравнения теплопроводности:

$$\frac{\partial T_{land}}{\partial t} = a_{land}\left(\frac{\partial^2 T_{land}}{\partial x^2} + \frac{\partial^2 T_{land}}{\partial y^2}\right), \quad (3)$$

$$\frac{\partial T_{Steel}}{\partial t} = a_{steel}\left(\frac{\partial^2 T_{steel}}{\partial x^2} + \frac{\partial^2 T_{steel}}{\partial y^2}\right), \quad (4)$$

В зоне соприкосновения земли с блоком стали действуют условия четвертого рода:

$$\begin{cases} \lambda_{land}\frac{\partial T_{land}}{\partial y} = \lambda_{steel}\frac{\partial T_{steel}}{\partial y} \\ T_{land} = T_{steel} \end{cases}, \quad (5)$$

где T_{land}, T_{steel} – температура земли и стали соответственно (°С), λ_{land}, λ_{steel} – коэффициенты теплопроводности земли и стали (Вт/м·К).

На верхней грани земли, где нет соприкосновения с блоком стали осуществляется теплообмен с окружающей средой, то есть действует граничное условие третьего рода:

$$\lambda_{land}\frac{\partial T_{land}}{\partial y} = \alpha_{air_1}(T_{land} - T_{air}), \quad (6)$$

где α_{air_1} – коэффициент теплообмена между землей и воздухом (Вт/м^2·К), T_{air} – температура окружающей среды (воздуха).

На всех гранях блока стали кроме нижней также осуществляется теплообмен с окружающей средой, то есть действуют граничные условия третьего рода:

$$\lambda_{steel}\frac{\partial T_{steel}}{\partial n} = \alpha_{air_2}(T_{steel} - T_{air}), \quad (7)$$

где α_{air_2} – коэффициент теплообмена между блоком стали и воздухом (Вт/м^2·К), T_{air} – температура окружающей среды (воздуха).

Температура земли в начальный момент времени равна $T_{land}(0, x, y) = 20$°С, а стали $T_{steel}(0, x, y) = 1500$°С. На боковых и нижней гранях земли действует мягкое граничное условие:

$$\frac{\partial T_{land}}{\partial n} = 0; \quad (8)$$

где n – нормаль к рассматриваемой грани.

В рассматриваемой задаче имеется плоскость симметрии, а, следовательно, будет достаточно провести расчеты лишь для одной из подобластей. Далее все рассуждения будут проводиться относительно правой подобласти. В силу существования плоскости симметрии на ней будут действовать условия симметрии для земли и блока стали соответственно:

$$\frac{\partial T_{land}}{\partial x} = 0, \quad (9)$$

$$\frac{\partial T_{steel}}{\partial x} = 0. \quad (10)$$

Далее рассмотрим дискретизацию и методы решения. Для этого рассмотрим область исследования в двумерной декартовой системе координат с центром, находящимся на пересечении правой границы области и границы соприкосновения блока стали и поверхности земли.

Для определения размеров областей введем следующие обозначения: $L_x \times L_y$ – размеры блока стали, $D_x \times D_y$ – размеры области земли. Область исследования покрывается единой декартовой сеткой со сгущением на блоке стали, как показано на рис. 2. Так, блок стали покрывается равномерной разностной сеткой с шагами h_x, h_y, а пласт земли сеткой с шагом H_y в направлении оси Oy и шагом h_x под блоком стали, так, чтобы на границе соприкосновения узлы совпадали. На оставшемся интервале от Lx до Dx используется сетка с более крупным шагом H_x (рис. 2).

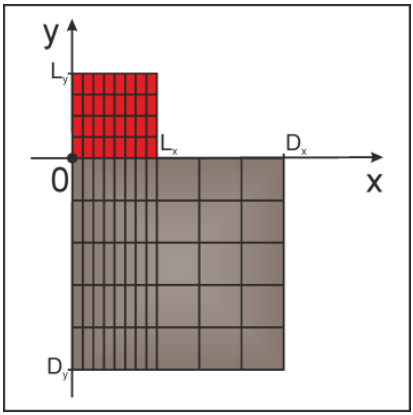


Рис. 2. Иллюстрация области исследования покрытой неравномерной сеткой

Численное решение поставленной задачи будем искать методом конечных разностей, с использованием аппроксимации на полушаге при переходе с одной сетки на другую [4]. Приведем конечно-разностные формулы для аппроксимации дифференциальных операторов [4]:

$$\left(\frac{\partial T}{\partial t}\right)_{i,j}^{n+1} \approx \frac{T_{i,j}^{n+1} - T_{i,j}^{n}}{\tau}, \quad (11)$$

$$\left(\frac{\partial^2 T}{\partial x^2}\right)_{i,j}^{n+1} \approx \frac{\lambda}{x_{i+\frac{1}{2}}-x_{i-\frac{1}{2}}}\left(\frac{T_{i+1,j}^{n+1}-T_{i,j}^{n+1}}{x_{i+1}-x_i}-\frac{T_{i,j}^{n+1}-T_{i-1,j}^{n+1}}{x_i-x_{i-1}}\right)+$$

$$+\frac{(1-\lambda)}{x_{i+\frac{1}{2}}-x_{i-\frac{1}{2}}}\left(\frac{T_{i+1,j}^{n}-T_{i,j}^{n}}{x_{i+1}-x_i}-\frac{T_{i,j}^{n}-T_{i-1,j}^{n}}{x_i-x_{i-1}}\right), \quad (12)$$

$$\left(\frac{\partial^2 T}{\partial y^2}\right)_{i,j}^{n+1} \approx \frac{\lambda}{y_{j+\frac{1}{2}}-y_{j-\frac{1}{2}}}\left(\frac{T_{i,j+1}^{n+1}-T_{i,j}^{n+1}}{y_{j+1}-y_j}-\frac{T_{i,j}^{n+1}-T_{i,j-1}^{n+1}}{y_j-y_{i-1}}\right)+$$

$$+\frac{(1-\lambda)}{y_{j+\frac{1}{2}}-y_{j-\frac{1}{2}}}\left(\frac{T_{i,j+1}^{n}-T_{i,j}^{n}}{y_{j+1}-y_j}-\frac{T_{i,j}^{n}-T_{i,j-1}^{n}}{y_j-y_{i-1}}\right), \quad (13)$$

где x_i, y_j – координаты узлов сетки, $x_{i+\frac{1}{2}}$, $x_{i-\frac{1}{2}}, y_{j+\frac{1}{2}}$, $y_{j-\frac{1}{2}}$ – координаты точек расположенных между соседними узлами сетки. Коэффициент λ определяет вид схемы ($0 \le \lambda \le 1$). При $\lambda = 0$ получается явная разностная схема, при $\lambda = 1$ – неявная разностная схема.

Явная и неявная разностные схемы аппроксимируют исходную систему дифференциальных уравнений с первым порядком по времени, со вторым порядком по координате на равномерной сетке и с первым порядком на неравномерной сетке: $O\left(\tau, H_x - h_x, {H_y}^2\right)$. В то время как неявная схема абсолютно устойчива, то явная схема имеет следующее условие устойчивости:

$$\tau\left(\frac{1}{\min(h_x^2, H_x^2)}+\frac{1}{\min(h_y^2, H_y^2)}\right) \le \frac{1}{2a}. \quad (14)$$

Полученные разностные схемы были исследованы на сеточную сходимость. На рисунке 3 представлено распределение температуры через 30 минут модельного времени на плоскости симметрии для четырех вариантов сеточного разрешения. Получено, что сеточную сходимость обеспечивает следующее минимальное сеточное разрешение $N_x = 19; N_y = 19; M_x = 40; M_y = 99$. Здесь N_x и M_x – количество внутренних точек при разбиении области исследования по оси Ox на отрезках $(0, L_x)$ и $(L_x, D_x]$ соответственно, N_y и M_y – количество внутренних точек разбиения по оси Oy для блока стали и пласта земли соответственно.

Разработка программной реализации поставленной задачи выполнена на языке программирования C++ с использованием библиотеки MPI при распараллеливании программы.

Распараллеливание алгоритма можно осуществить по данным или по вычислениям. В рассматриваемой задаче использовано распараллеливание по данным, когда производится разрезание области исследования на части и на каждой подобласти вычисления производятся одновременно на разных процессах. Также разрезание двумерной области может быть реализовано с помощью одномерной или двумерной декомпозиции. В расчетах будет использоваться одномерная декомпозиция расчетной области (рис. 4, *а*).

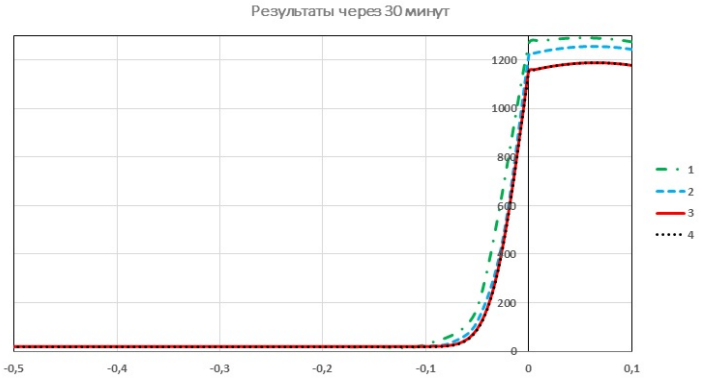


Рис. 3. Сеточная сходимость

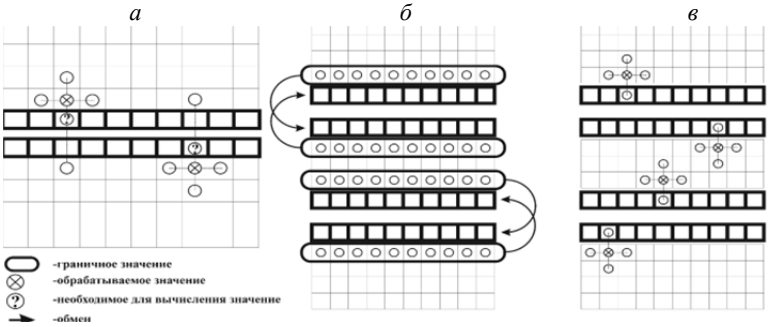


Рис. 4. Схема декомпозиции

В силу используемого шаблона разностной схемы, при расчете в приграничных узлах сетки на каждом процессе будут появляться потреб-

ность в использовании данных с соседнего процесса, как показано на рис. 4, *а*. Потому создадим фиктивные ячейки на каждом процессе и организуем обмен данными между соседними процессами. То есть приграничные значения с соседних процессов будут пересылаться в фиктивные ячейки (рис. 4, *б*) [1, 2].

Для обмена данными между процессами использовались процедуры: MPI_Send и MPI_Recv (отправить сообщение и принять сообщение соответственно). После окончания расчетов результаты вычислений собирались на процессе с номером 0. Для этого использовалась процедура MPI_Gather.

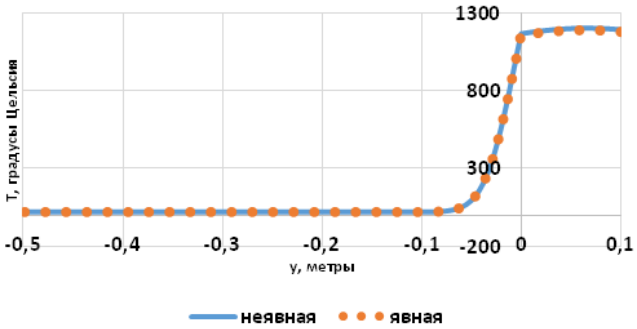


Рис. 5. Графики сравнения температур на оси *Oy* через 30 минут

Для сравнения решений, полученных с использованием явной и неявной разностных схем, проведено несколько вычислительных экспериментов. На рис. 5 представлено распределение температуры на линии симметрии спустя 30 мин модельного времени для случая использования явной и неявной разностных схем. Видно, что полученные результаты совпадают.

Для верификации разработанных программ проведено решение этой же задачи в программе ANSYS Fluent с использованием тех же значений констант и тех же параметров сетки. В качестве области сравнения результатов также использовалась плоскость симметрии. Уже было показано, что решения, полученные явной и неявной разностными схемами, идентичны, поэтому достаточно сравнить результаты ANSYS Fluent с решением, полученным в случае использования явной разностной схемы.

На рис. 6 представлены результаты расчетов, полученные в ANSYS Fluent и полученные в разработанной программной реализации явной

разностной схемы спустя 6 и 40 часов модельного времени. Как видно из графиков, решения, полученные разными программами, показывают одинаковое распределение температуры вдоль оси симметрии, из чего можно сделать вывод, что решение задачи было проведено верно.

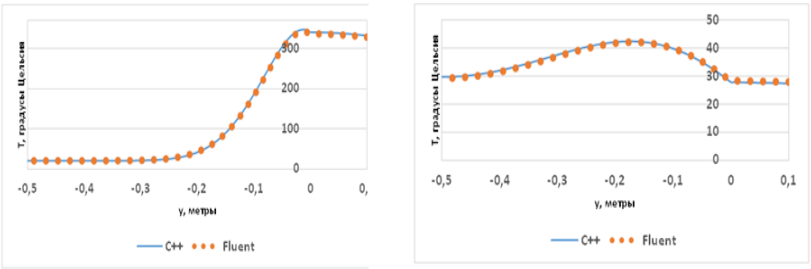


Рис. 6. Графики сравнения температур на оси Oy через 6 и 40 часов

Таким образом, в данной работе рассматривалось численное решение задачи распределения температуры в системе тел с различными теплофизическими свойствами. В ходе работы было предложено несколько способов решения поставленной задачи, а именно: численное решение с использованием явной и неявной схемой. Был проведен анализ аппроксимации, устойчивости и сходимости предложенных схем. Полученные алгоритмы были реализованы для МВС с распределенной памятью с использованием библиотеки MPI и языка программирования C.

Показано, что решения, полученные явной и неявной разностными схемами, совпадают. Также показано, что результаты численного решения полученного в разработанных программных реализациях совпадают с решением, полученным в программе ANSYS Fluent.

Литература

1. Данилкин Е.А. Распараллеливание явных и неявных разностных схем, эффективность параллельных и последовательных программ. Томск : Изд-во Том. ун-та, 2014. 41 с.
2. Старченко А.В. Берцун В.Н. Методы параллельных вычислений. Томск : Изд-во Том. ун-та, 2013. 225 с.
3. Кузнецов Г.В. Шеремет М.А. Разностные методы задач теплопроводности. Томск : Изд-во Том. политех. ун-та, 2007. 172 с.
4. Самарский А.А. Теория разностных схем. М., 1977. 656 с.

DOI: 10.17223/9785946216531/7

Опыт освоения технологий параллельного программирования в Томском университете систем управления и радиоэлектроники (ТУСУР)

Н.П. Фефелов

Кафедра автоматизированных систем управления, ТУСУР, Томск
fnp38@mail.ru

Дисциплина «Параллельное программирование» была впервые включена в учебный план подготовки инженеров по специальности 2204 – «Программное обеспечение вычислительной техники и автоматизированных систем» по государственному образовательному стандарту 1995 г. Преподавание ее ведется на кафедре АСУ с 1998 г. По мере совершенствования ГОСов место дисциплины, ее содержание и объем менялись.

В настоящее время дисциплина входит в учебный план подготовки бакалавров по направлению 09.03.01 – «Информатика и вычислительная техника», профиль – «Программное обеспечение вычислительной техники и автоматизированных систем». Она включена в цикл специальных дисциплин как вариативная дисциплина вузовского компонента и изучается в 7 семестре.

Виды занятий (часов): лекции – 18, лабораторные – 36, самостоятельная работа – 58. Зачет.

Параллельное программирование как учебная дисциплина в основном развивается в университетах на специальностях прикладной математики. Наше мнение – для инженеров-программистов следует больше уделять внимание технологическим вопросам разработки параллельных программ, меньше – типовым алгоритмам.

Основная нагрузка приходится на лабораторные занятия. В них осваиваются наиболее используемые технологии MPI и OpenMP.

Лабораторные работы выполняются на небольшом кластере (9 узлов, 72 ядра) кафедры АСУ в среде ОС Linux. Обеспечен доступ к кластеру по сети из любого места, где есть интернет.

Темы лабораторных работ:

1. Обработка и выполнение модульных программ в операционной системе UNIX.

2. Основные функции MPI.

3. Коллективные функции в MPI. Обработка массивов.

4. Режимы передачи данных в MPI.

5. Производные типы данных в MPI.

6. Обработка и выполнение программ в среде OpenMP.

7. Обработка массивов с использованием MPI и OpenMP.

8. Синхронизация потоков в OpenMP.

По каждой работе оформляется и защищается текстовый отчет, оформленный по правилам стандарта ТУСУРа.

1. В первой лабораторной работе студенты знакомятся с правилами выполнения лабораторных работ в сети кафедры АСУ, проводят индивидуальную настройку связных программ Putty и WinSCP. В практической части осваивается применение компилятора gcc в разных режимах при выполнении индивидуального задания по сборке исполняемой программы из нескольких модулей, заданных в исходном и объектном представлении. Требуется пятью вызовами gcc получить ассемблерное представление математической функции, объектное представление исходных модулей, скомпоновать программу и наконец, выполнить всю работу создания исполняемой программы одной командой gcc.

2. Шесть основных функций MPI изучаются на примере параллельной программы интегрирования. Студент получает одну из двух заданных программ в исходном виде и индивидуальное задание для подынтегральной и первообразной функций двух переменных.

Этапы лабораторной работы:

– получить загрузочную программу для заданного варианта программы и проверить ее в последовательном и параллельном режиме;

– заменить математические функции программы индивидуальными функциями, обработать программу и выполнить ее на разном числе процессов. Отмечать время выполнения;

– заменить индивидуальные функции передачи и приема на коллективные функции MPI_Bcast и MPI_Reduce. Провести тестирование программы;

– организовать ввод пределов интегрирования и параметра функции с терминала в нулевом процессе. Для передачи данных другим процессам использовать функции MPI_Pack и MPI_Unpack.

3. В работе с коллективными функциями используются MPI_Scatterv и MPI_Gatherv для обработки массивов. Дается образец параллельной программы. Для равномерного распределения элементов вспомогательных массивов для числа и смещения от начала исходного массива используется ал-

горитм добавления единицы из остатка от деления числа элементов исходного массива на число процессов в первые элементы вспомогательных массивов. Порции массивов для каждого ранга задаются динамически.

Для индивидуальных заданий используются алгоритмы обработки векторов и матриц:

– скалярное произведение;
– статистическая обработка;
– все частные суммы вектора;
– вычисление полинома по схеме Горнера;
– циклическая редукция;
– обработка строк матрицы;
– и другие.

4. В работе «Режимы передачи данных в MPI» берется программа вычисления интеграла для индивидуальной функции с шестью основными функциями MPI. Блокирующие функции передачи заменяются на функции, указанные в индивидуальном задании. Даются примеры использования функций без блокировки, с буферизацией, правила использования управляющих параметров request и status и функций семейства MPI_Wait и MPI_Test. Проводится сравнительное выполнение программ с оценкой времени выполнения на различном числе процессов.

5. В работе «Производные типы данных» необходимо сконструировать новый тип данных для представления части матрицы. Используются векторный или индексный способы конструирования. Необходимо выбрать наиболее подходящий способ конструирования в соответствии с индивидуальным заданием. Составить алгоритм вычисления аргументов конструктора производного типа и заполнить массивы размера блоков и их смещений этим алгоритмом (для индексного способа). После регистрации производного типа дважды передать в другой процесс выбранные элементы матрицы как единственный элемент сконструированного типа. Второй процесс принимает первую посылку в матрицу размера исходной матрицы. Прием второй передачи воспринимает полученные данные как одномерный массив типа базовых элементов исходной матрицы.

В индивидуальных заданиях предлагается выбирать разреженные строки или столбцы матрицы, составлять подматрицы, выбирать различные треугольные части, конструировать шахматную доску, выбирать диагональные элементы и т.д.

Пример выходных данных для производного типа – диагональный крест в рамке.

Rank 1. Output array.

1 2 3 4 5 6 7
8 9 0 0 0 **13 14**
15 0 **17** 0 **19** 0 **21**
22 0 0 **25** 0 0 **28**
29 0 **31** 0 **33** 0 **35**
36 37 0 0 0 **41 42**
43 44 45 46 47 48 49

Rank 1. Subarray.

1 2 3 4 5 6 7 8 14 15 21
22 28 29 35 36 42 43 44 45 46 47
48 49 9 13 17 19 25 31 33 37 41

6. Использование технологии OpenMP начинается с использования программы вычисления интеграла. Сначала убираются функции MPI и делается последовательная программа. Затем планируются параллельные регионы программы, определяется набор общих и локальных переменных, назначается переменная для редукции. В программу добавляются директивы parallel и for с необходимыми параметрами и типами переменных. Программа тестируется в статическом (static) и динамическом (dynamic) режимах управления циклом. Число нитей для выполнения программы задается в командной строке при запуске программы.

7. «Обработка массивов с использованием MPI и OpenMP». Программа строится на основе ранее созданной программы обработки массивов. Распараллеливается обработка части массива в каждом процессе директивами parallel и for. Планируются итерации цикла в параллельной области (schedule). В параллельной области операциями редукции определяются итоговые данные (если они есть) для каждого процесса. Функцией MPI_Reduce формируются глобальные результаты.

8. Параллелизм задач осваивается на секциях OpenMP в лабораторной «Синхронизация потоков в OpenMP». Индивидуальные здания предлагают составить программу для одной из типичных проблем: алгоритмы Деккера и Петерсона, производитель-потребитель, читатели-писатели, обедающие философы, спящий брадобрей. Шаблоны программ, взятые из книг, даются в руководстве. Для каждого варианта задания указываются конкретные средства синхронизации из набора: семафоры и мьютексы, условные переменные, директивы atomic, критические секции. Для сравнения выполняется программа без средств синхронизации.

К сожалению, на кафедре пока нет оборудования для параллельных вычислений на GPU – в программной модели CUDA.

Лекции носят обзорный характер. Темы: состояние и проблемы параллельных вычислений, моделирование и анализ параллельных алгоритмов, интерфейс передачи сообщений MPI, технология программирования OpenMP, технология программирования CUDA и OpenACC.

Подготовлены презентации для иллюстрации основных принципов параллелизма на основе широко распространенных курсов в интернете В.П. Гергеля, А.С. Антонова, А.В. Борескова и других авторов. Файлы лекционных презентаций и описаний лабораторных работ хранятся в общих каталогах в сети кафедры и всегда доступны. В библиотеке вуза имеются книги серии «Суперкомпьютерное образование». Однако поколение next часто выбирает интернет.

DOI: 10.17223/9785946216531/8

Обработка изображений с помощью Shearlets

А.А. Потоцкая, А.А. Захарова

Томский государственный университет, Томск
Университет Руана, Руан, Франция
bubuzyonok@yandex.ru

В статье дано определение и основные возможности для приложений Shearlets. Рассматривается проблема удаления шума с изображения с помощью вейвлетов и Shearlets. Проводится сравнение полученных результатов.

Ключевые слова: обработка изображений, вейвлеты, Shearlets.

В последнее время постоянно возрастает роль цифровых данных. Это связано с увеличивающимся распространением различной цифровой техники и, как следствие, значительным ростом объема данных, которые необходимо хранить и обрабатывать, что требует больших компьютерных ресурсов. Поэтому задача сокращения объема хранимой и обрабатываемой информации за счет отбрасывания несущественных частей имеет большую значимость в наши дни.

Разработка эффективных (с точки зрения использования компьютерных ресурсов) алгоритмов обработки больших потоков информации ведется еще с прошлого века. Основной идеей всех этих методов является разложение исходного потока информации на три менее плотных потока: основной информационный поток, уточняющий информационный поток и поток с несущественной информацией. Так как основной информационный поток гораздо менее плотный, чем исходный, то его можно передать легче и быстрее. Уточняющий информационный поток не всегда необходим, поток с несущественной информацией вообще может не учитываться. Этот же принцип используется для избавления от шумов в информационных сигналах.

Для подобной обработки информации используются сложные математические и численные методы. Среди них можно выделить вейвлеты.

Вейвлеты были очень популярны в последние десятилетия прошлого века. Этому были две основные причины. Во-первых, многие люди были заинтересованы в данной теме, так как вейвлеты – это комплекс идей, базирующихся в разных областях науки, таких как инженерия, физика и

чистая математика. Во-вторых, вейвлеты – это простой математический инструмент с большим количеством возможностей для приложений. Дополнительный аспект популярности теории вейвлетов – это их богатая математическая структура, позволяющая создавать семейства вейвлетов с различными желаемыми свойствами. Следовательно, вейвлеты стали своего рода революцией в области обработки изображений и сигналов и позволили создать многочисленные крайне успешные приложения, в том числе алгоритм JPEG2000 - современный стандарт сжатия изображений.

Развитие теории вейвлетов началось с работы Хаара в начале 20 века. Однако термин «вейвлет» появился гораздо позже. Значительный вклад в теорию вейвлетов принадлежит Морлету, Гроссманну, Маллату [1–3]. Первые работы, обобщающие идеи вейвлетов, увидели свет только в 90х годах XX в. Среди таких работ можно назвать работы Чуи и Добеши [4, 5]. В настоящее время анализ с помощью вейвлетов – это один из мощнейших и в то же время гибких инструментов для обработки информации. По этой теме существует большое количество публикаций, а также данный вид анализа встречается в таких средах математического программирования как MATLAB, Maple, Matematica.

Функция $\psi \in L_2(\mathbb{R})$ называется ортогональным вейвлетом, если семейство $\{\psi_{j,k}\}$ определенное как:

$$\psi_{j,k}(x) := 2^{j/2}\psi(2^j x - k), \qquad j,k \in \mathbb{Z}, \tag{1}$$

является ортонормированным базисом в $L_2(\mathbb{R})$, то есть

$$< \psi_{j,k}, \psi_{l,m} >= \delta_{j,l}\delta_{k,m}, \qquad j,k,l,m \in \mathbb{Z} \tag{2}$$

и каждая функция $f \in L_2(\mathbb{R})$ может быть записана в виде:

$$f(x) = \sum_{j,k=-\infty}^{+\infty} C_{j,k}\psi_{j,k}(x). \tag{3}$$

Ряды в представлении функции f в виде (3) называются «вейвлет-рядами». Вейвлет-коэффициенты $C_{j,k}$ даны следующей формулой:

$$C_{j,k} =< f, \psi_{j,k} >. \tag{4}$$

Несмотря на свою популярность, вейвлеты не слишком эффективны при работе с многомерными данными. Вообще говоря, вейвлеты полезны только при работе с точечными сингулярностями. В более высоких измерениях не только встречаются, но и преобладают другие типы сингулярностей, с которыми вейвлеты не способны справиться. Как следствие, вейвлеты не являются оптимальным средством для работы с многомерными данными.

Для того чтобы преодолеть этот недостаток традиционных вейвлетов, необходимо увеличить их чувствительность к направлению воздействия и для этого в последние годы было предложено несколько методов. Большой прорыв произошел с введением «curvelets» Кандесом и Донохо в 2004. Это была первая система, предоставляющая оптимально разреженные аппроксимации для класса двумерных функций, проявляющих анизотропные свойства. Именно поэтому curvelets хорошо подходят для использования в качестве адаптивной системы представления с точки зрения их способности сближать изображения границами. У curvelets есть два основных дефекта. Во-первых, curvelets не порождаются путем воздействия конечного семейства операторов на единственную функцию. Во-вторых, их конструкция включает в себя вращения, что исключает прямой переход от непрерывных данных к дискретным сигналам.

«Contourlets» были представлены в 2005 году До и Веттерли как отдельно существующий набор фильтров на основе curvelets. Этот подход предполагает применение древовидного фильтра подобного стандартному, использованному для получения крайне успешных численных алгоритмов. Однако теория неразрывности в данном методе теряется.

В том же году «Shearlets» [6] были представлены Гуо, Кутынёк, Лабате, Лим и Вайссом. Данный метод предусматривал более широкий класс родственных систем, так называемых «композитных вейвлетов», и их использование в качестве многомерных расширений на базе вейвлетов.

Одна из специфических характеристик Shearlets - это использование сдвига для управления пространственной направленностью, в отличие от curvelets, которые вращаются. Это в корне отличающаяся концепция, так как она позволяет системе Shearlets дифференцироваться от единичного или конечного набора порождающих функций, что также позволяет Shearlets обобщить концепцию непрерывного и дискретного мира. Таким образом, представление Shearlets предлагает уникальную комбинацию следующих условий:

- Единичный или конечный набор порождающих функций.
- Оптимально разреженные аппроксимации анизотропных свойств многомерных данных.
- Анализирующие элементы на компактных носителях.
- Быстрое алгоритмическое применение.
- Обобщенное рассмотрение непрерывных и дискретных областей.
- Связь с классическими аппроксимативными пространствами.

Рассмотрим общие идеи построения Shearlet-систем. Как мы уже видели в случае системы вейвлетов, анализирующие элементы должны состоять из сигналов, изменяющихся в нескольких масштабах, ориентациях и местоположениях с возможностью растягиваться. Поэтому необходимо объединить оператор масштабирования, ортогональный оператор и оператор переноса.

Масштабирующий оператор необходим для порождения элементов в разных масштабах, более того, масштабирующий оператор должен порождать волны с анизотропным носителем, поэтому будем использовать семейство операторов дилатации $D_{A_a}, a > 0$:

$$D_{A_a}\psi(x) = |\det A_a|^{-1/2}\psi\left(A_a{}^{-1}x\right). \quad (5)$$

где A_a – масштабирующая матрица:

$$A_a = \begin{pmatrix} a & 0 \\ 0 & a^{\alpha} \end{pmatrix}, \quad (6)$$

с параметром $\alpha \in (0,1)$, который регулирует «степень анизотропии». В данной работе будем рассматривать случай $\alpha = 1/2$, который является оптимальным для дискретизации параметров Shearlet-систем.

Оператор ортогонального преобразования необходим для изменения ориентации волн. Наиболее очевидным выбором в данном случае является оператор вращения. Однако, вращения могут разрушить структуру пространства $\mathbb{Z}^2$, что оказывается серьезной проблемой при переходе от непрерывных областей к дискретным. Поэтому в качестве оператора ортогонального преобразования используется оператор сдвига $D_{S_s}, s \in \mathbb{R}$, где матрица сдвига имеет вид:

$$S_s = \begin{pmatrix} 1 & s \\ 0 & 1 \end{pmatrix}. \quad (7)$$

Оператор переноса необходим для перемещения волн по двухмерной поверхности. Для этого используется стандартный оператор T_t, определяющийся по формуле:

$$T_t\psi(x) = \psi(x - t). \quad (8)$$

Совмещая данные три оператора, мы можем определить непрерывные Shearlet-системы [7].

Пусть $\psi \in L^2(\mathbb{R}^2)$, тогда непрерывная Shearlet-система - это

$$SH(\psi) = \left\{\psi_{a,s,t} = T_t D_{A_a} D_{S_s}\psi : a > 0, s \in \mathbb{R}, t \in \mathbb{R}^2\right\}. \quad (9)$$

Shearlet-функции были введены для решения ряда проблем при представлении и обработке многомерных данных, и они были успешно использованы в нескольких числовых приложениях.

Приложения Shearlets для обработки изображений. Использование Shearlets выгодно для различных проблем восстановления данных и выделения признаков. В частности, один из видов приложений, с которыми Shearlets справились весьма успешно, – это проблемы с шумоподавлением изображений. Были предложены несколько алгоритмов шумоподавления изображений на основе Shearlets, в том числе методы [8, 9], которые адаптируют бинаризацию с помощью вейвлетов к Shearlets-постановке, метод [10], который сочетает бинаризацию с минимизацией ограниченной вариации.

Приложения Shearlets для разделения данных. В нескольких практических приложениях важно разделить данные на подкомпоненты. Например, в астрономических изображениях необходимо отделять звезды от галактик. То есть необходимо разделить точечные и кривоподобные структуры. Используя методы разреженной аппроксимации и комбинируя разложения вейвлетов и Shearlets, в [11, 12] был разработан очень эффективный метод разделения данных.

Приложения Shearlets для решения обратных задач. Методы на основе Shearlets также применялись для построения регуляризованного алгоритма инверсии для преобразования Радона. Это преобразование лежит в основе компьютерной томографии [13]. Было также показано, что аналогичные идеи полезны при работе с более общими классами обратных задач, такими как деблокирование и деконволюция [14].

Рассмотрим задачу удаления шума с изображения. Теория удаления шумов с помощью вейвлет-обработки уже достаточно хорошо развита. Кроме того, известно, что именно вейвлет-обработка является оптимальным методом устранения шума для сигналов с заранее не известной формой, обладающих определенной степенью гладкости. Однако, как уже было отмечено выше, вейвлеты имеют существенные недостатки при работе с многомерными данными.

Проведем численное сравнение методов подавления шума с помощью вейвлет-обработки и Shearlet-обработки.

В качестве критерия сравнения двух методов будем использовать пиковое отношение сигнала к шуму, то есть соотношение между максимумом возможного значения сигнала и мощностью шума, искажающего значения сигнала.

Очевидно, что в случае двумерного сигнала (изображения) Shearlet-обработка лучше подходит для устранения шума, чем вейвлет-обработка. Это подтверждается и значениями пиковых отношений сигнала к шуму: для вейвлет-обработки – 25,1219, для Shearlet-обработки – 33,7668.

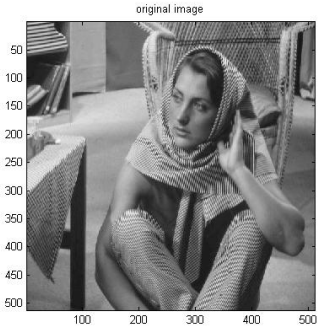


Рис. 1. Оригинальное изображение

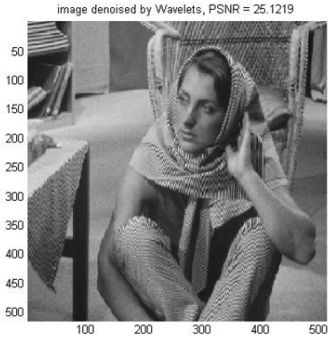


Рис. 2. Обработка с помощью вейвлетов

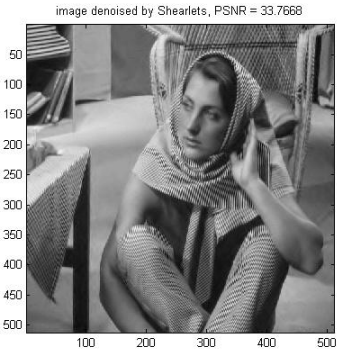


Рис. 3. Обработка с помощью Shearlets

Литература

1. Morlet J. Sampling theory and wave propagation //Issues in Acoustic Signal – Image Processing and Recognition. Springer Berlin Heidelberg, 1983. P. 233–261.
2. Kronland-Martinet R., Morlet J., Grossmann A. Analysis of sound patterns through wavelet transforms // International journal of pattern recognition and artificial intelligence. 1987. V. 1, No. 02. P. 273–302.
3. Mallat S. G. A theory for multiresolution signal decomposition: the wavelet representation // IEEE transactions on pattern analysis and machine intelligence. 1989. V. 11, No. 7. P. 674–693.
4. Chu C. K. An introduction to wavelets. New York, NY, USA: Academic, 1992.
5. Daubechies I. Ten lectures on wavelets. Society for industrial and applied mathematics. 1992.
6. Guo K., Kutyniok G., Labate D. Sparse multidimensional representations using anisotropic dilation and shear operators. 2006.
7. Kutyniok G., Labate D. Introduction to shearlets // Shearlets. Birkhäuser Boston, 2012. P. 1–38.
8. Easley G., Labate D., Lim W. Q. Sparse directional image representations using the discrete shearlet transform // Applied and Computational Harmonic Analysis. 2008. V. 25, No. 1. P. 25–46.
9. Lim W. Q. The discrete shearlet transform: A new directional transform and compactly supported shearlet frames // IEEE Transactions on Image Processing. 2010. V. 19, No. 5. P. 1166–1180.
10. Han B., Kutyniok G., Shen Z. Adaptive multiresolution analysis structures and shearlet systems // SIAM Journal on Numerical Analysis. 2011. V. 49, No. 5. P. 1921–1946.
11. Donoho D., Kutyniok G. Geometric separation using a wavelet-shearlet dictionary // SAMPTA'09. 2009. C. Special session on geometric multiscale analysis.
12. Kutyniok G., Lim W. Q. Image separation using wavelets and shearlets // Curves and Surfaces: 7th International Conference, Avignon, France, June 24–30, 2010, Revised Selected Papers. Springer Science & Business Media, 2011. 416 p.
13. Colonna F. et al. Radon transform inversion using the shearlet representation // Applied and Computational Harmonic Analysis. 2010. V. 29, No. 2. P. 232–250.
14. Patel V.M., Easley G.R., Healy D.M. Shearlet-based deconvolution // IEEE Transactions on Image Processing. 2009. V. 18, No. 12. P. 2673–2685.

DOI: 10.17223/9785946216531/9

Численное исследование течения в русле реки во время весеннего ледохода для определения возможных зон локальных подтоплений*

В.В. Чуруксаева, А.В. Старченко

Томский государственный университет, Томск
chu.vv@mail.ru

Работа посвящена моделированию течения в русле реки во время весеннего ледохода. Для решения поставленной задачи была построена новая математическая модель нестационарного двухфазного изотермического движения смеси «вода – ледяные частицы» в открытом русле (канале) на основе уравнений механики взаимодействующих взаимопроникающих континуумов. Разработан численный метод. Проведено численное исследование течения на участке р. Томь в момент ее вскрытия ото льда.
Ключевые слова: двухфазное течение, приближение мелкой воды, течение в русле реки, ледяные частицы, вскрытие реки ото льда.

Процессы замерзания и вскрытия рек, а также формирования заторов из крупных льдин сложно поддаются прогнозированию из-за трудности выполнения полевых исследований и потому весенние паводки, связанные с таянием снега и вскрытием реки ото льда часто являются причиной чрезвычайных ситуаций, приводящих к наводнениям в прибрежных городах и особенно небольших поселках [1].

Существует два класса математических моделей течения реки во время ледохода: статические модели и динамические модели. Классические статические модели [2,3] являются достаточно грубым описанием течения в реке. Такой подход позволяет оценить толщину льда в уже образовавшемся заторе, но не позволяет определить, появится ли ледовый затор, возможные зоны его локализации и время образования. Кроме того, проведение расчета по статическим моделям требует наличия подробных данных о физических свойствах объекта моделирования [4]. Именно статические модели заложены в современные пакеты прикладных программ для расчета речных течений.

* Работа выполнена при поддержке РФФИ и Администрации Томской области в рамках научного проекта № 16-41-700178 р_а.

Современные динамические модели ледовых процессов в реках [5–8] предполагают построение моделей на основе многомерных уравнений гидродинамики, которые позволяют прогнозировать время и место возможного появления ледовых заторов.

Физическая постановка задачи

В данной работе в качестве приближенного представления течения реки во время ледохода рассматривается двухфазное изотермическое движение смеси «вода – легкие частицы» в открытом канале или русле реки. Межфазовый обмен массой и теплом не учитывается в силу близких значений температуры воды и окружающей среды и их незначительных изменений за период моделирования. Плотность льда $\rho_i^0 = 910$ кг/м^3 меньше плотности воды $\rho_l^0 = 1000$ кг/м^3 и потому считается, что ледяные частицы плотно расположены в приповерхностном слое воды и их концентрация остается постоянной на входе в канал (или рассматриваемый участок реки).

Предполагается, что горизонтальные размеры области исследования много больше глубины двухфазного потока и при этом размер ледяных частиц много меньше характерных размеров канала (русла).

Для математического описания данного процесса будем использовать уравнения механики взаимодействующих взаимопроникающих континуумов [9].

Сделанные выше предположения позволяют рассматривать фазу ледяных частиц в качестве непрерывной сплошной среды с эффективными свойствами. В силу предположения о существенном различии масштабов рассматриваемого процесса по горизонтали и вертикали будем использовать гидростатическое приближение.

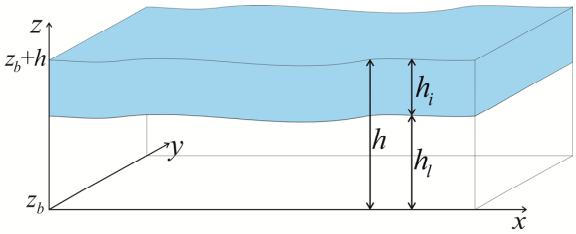


Рис. 1. Область исследования

Предполагая, что распределение динамических параметров двухфазного течения по глубине близко к однородному, будем интегрировать

записанные уравнения движения фаз по всей глубине потока (от z_b до $h+z_b$). В результате получим следующие осредненные по глубине уравнения неразрывности

$$\frac{d_l h'}{dt} = 0;$$

и движения

$$\frac{d_l h'\overline{u}_l}{dt} = -gh'\frac{\partial\left(z_b+h\right)}{\partial x} + \frac{\partial}{\partial x}\left(h'\tau_{xxl}\right) + \frac{\partial}{\partial y}\left(h'\tau_{xyl}\right) + \mathrm{S}_{lx},$$

$$\frac{d_l h'\overline{v}_l}{dt} = -gh'\frac{\partial\left(z_b+h\right)}{\partial y} + \frac{\partial}{\partial x}\left(h'\tau_{yxl}\right) + \frac{\partial}{\partial y}\left(h'\tau_{yyl}\right) + \mathrm{S}_{ly}$$

для несущей фазы и соответствующими уравнениями

$$\frac{d_i h''\overline{u}_i}{dt} = 0;$$

$$\frac{d_i h''\overline{u}_i}{dt} = -\frac{\rho_l^0}{\rho_i^0}gh''\frac{\partial\left(z_b+h\right)}{\partial x} + \frac{\partial}{\partial x}\left(h''\tau_{xxi}\right) + \frac{\partial}{\partial y}\left(h''\tau_{xyi}\right) + \mathrm{S}_{ix},$$

$$\frac{d_i h''\overline{v}_i}{dt} = -\frac{\rho_l^0}{\rho_i^0}gh''\frac{\partial\left(z_b+h\right)}{\partial y} + \frac{\partial}{\partial x}\left(h''\tau_{xyi}\right) + \frac{\partial}{\partial y}\left(h''\tau_{xxi}\right) + \mathrm{S}_{iy}$$

для дисперсной фазы ледяных частиц.

Здесь $\frac{d\Phi}{dt} = \frac{\partial\Phi}{\partial t} + u\frac{\partial\Phi}{\partial x} + v\frac{\partial\Phi}{\partial y}$; h – глубина всего потока; $0 \le \overline{\alpha}_l \le 1,\ 0 \le \overline{\alpha}_i \le 1,\ \overline{\alpha}_l + \overline{\alpha}_i = 1$ – массовые доли льда и воды; $h'' = \overline{\alpha}_i h_i$; h_i – глубина слоя дисперсной фазы; $h' = h - h''$ – глубина слоя несущей фазы, где $\overline{\alpha}_i = 0$; $\vec{\overline{w}}_l = \left(\overline{u}_l, \overline{v}_l\right)$ – осредненная по глубине скорость несущего слоя, $\vec{\overline{w}}_i = \left(\overline{u}_i, \overline{v}_i\right)$ – осредненная по толщине слоя скорость дисперсной фазы; g – ускорение свободного падения; ρ_l^0, ρ_i^0 – истинные плотности воды и льда соответственно. Источниковые члены $\vec{\mathrm{S}}_l, \vec{\mathrm{S}}_i$ включают в себя силу плавучести, силу вязкого трения, силу присоединенной массы, силу Кориолиса и силу трения о дно [9]. Данная модель подробно представлена в [10].

Для расчета турбулентных характеристик двухфазного течения используется $k-\varepsilon$ модель турбулентности для осредненных уравнений с

модификацией Поурахмади и Хумфри [11] для учета влияния дисперсных частиц.

Уравнения модели имеют вид:

$$\frac{d_l h'\overline{k}}{dt} = \frac{\partial}{\partial x}\left[h'\left(\nu + \frac{\overline{\nu}_t}{\sigma_k}\right)\frac{\partial \overline{k}}{\partial x}\right] + \frac{\partial}{\partial y}\left[h'\left(\nu + \frac{\overline{\nu}_t}{\sigma_k}\right)\frac{\partial \overline{k}}{\partial y}\right] + \left(P_k + P_{kv} - \overline{\varepsilon}\right)h' - \overline{\varepsilon}_P h_i;$$

$$\frac{d_l h'\overline{\varepsilon}}{d_l t} + \frac{\partial h'\overline{u}_l\overline{\varepsilon}}{\partial x} + \frac{\partial h'\overline{v}_l\overline{\varepsilon}}{\partial y} = \frac{\partial}{\partial x}\left[h'\left(\nu + \frac{\overline{\nu}_t}{\sigma_\varepsilon}\right)\frac{\partial \overline{\varepsilon}}{\partial x}\right] + \frac{\partial}{\partial y}\left[h'\left(\nu + \frac{\overline{\nu}_t}{\sigma_k}\right)\frac{\partial \overline{\varepsilon}}{\partial y}\right] +$$

$$+\left(c_1\frac{\overline{\varepsilon}}{\overline{k}}P_k + P_{\varepsilon v} - c_2\frac{\overline{\varepsilon}^2}{\overline{k}}\right)h' - \frac{\varepsilon_P\overline{\varepsilon}}{\overline{k}}h_i;$$

$$\overline{\nu}_t = c_\mu\frac{\overline{k}^2}{\overline{\varepsilon}}.$$

Здесь $\overline{k}(t,x,y)$ – осредненная по глубине кинетическая энергия турбулентности; $\overline{\varepsilon}(t,x,y)$ – осредненная по глубине диссипация кинетической энергии;

$\overline{k}_i = \overline{k}\dfrac{T_L}{\tau_i + T_L}$ – осредненная по глубине кинетическая энергия в слое дисперсной фазы;

$\overline{\varepsilon}_P = \dfrac{\rho_i^0\overline{\alpha}_i 2\overline{k}}{\left(\tau_i + T_L\right)\rho_l^0}$ – дополнительная диссипация кинетической энергии за счет дисперсной фазы;

$T_L = 0,41\dfrac{\overline{k}}{\overline{\varepsilon}}$ – лагранжев масштаб турбулентности;

$\tau_i = \dfrac{\rho_i^0 d_i^0}{18\mu_l}$ – время релаксации частиц;

$$\overline{\nu}_{it} = \overline{\nu}_t\frac{\rho_l^0}{\rho_i^0}\left(1 - \frac{\alpha_l T_L\overline{\varepsilon}}{5\nu_l}\frac{\Omega^2}{\Omega+2}\right);\ \Omega = \frac{2\tau_i}{T_L}.$$

Константы модели выбирались аналогично стандартной высокорейнольдсовой $k-\varepsilon$ модели турбулентности.

Начальные и граничные условия. В начальный момент времени $t = 0$ используются следующие значения параметров течения:

$h'' = h_{ice}$ – известная величина;

$\overline{u}_i = \overline{u}_{i0}$, $\overline{v}_i = \overline{v}_{i0}$;

$h' = h - h_{ice}$;

$\overline{u}_l = \overline{u}_{l0}$, $\overline{v}_l = \overline{v}_{l0}$.

На входе в расчетную область значения параметров жидкой фазы и фазы частиц считаются известными, на выходной границе используется равенство нулю производных по внешней нормали к границе. Кроме того, на границах потока с берегом рассматривается трение как для жидкости, так и для частиц. На боковых стенках русла применяются условия непротекания и прилипания для компонент скорости. В случае, когда вблизи боковых стенок в несущей фазе справедливо соотношение $\nu_{l\,t} \gg \nu_l$, трение и турбулентные характеристики в несущей фазе в пристеночной области определяются с помощью метода пристеночных функций Лаундера-Сполдинга [12]. Трение фазы частиц о дно реки в прибрежной зоне и на участках отмелей (при условии $h'' > \frac{h'}{2}$) выражается зависимостью $c_f^i \left|\vec{\mathbf{w}}_\mathbf{i}\right| \vec{\mathbf{w}}_\mathbf{i}$, где $c_f^i = 0.0025 \left|\vec{w}_i\right|^{-1}$ [13].

Численный метод

Для построения численной модели уравнения модели аппроксимировались со вторым порядком по пространству методом конечного объема на разнесенной структурированной сетке. Для исключения из расчета сухих ячеек применялся метод фиктивных областей.

Численный метод решения уравнений модели основан на явно-неявной аппроксимации по времени и алгоритме исключения неизвестных, позволяющем использовать построенную разностную схему для областей двухфазного течения, где отсутствует дисперсная фаза, и существенно ослабить ограничение на шаг интегрирования по времени для уравнений движения.

Построенная математическая модель была протестирована на расчетах течения в открытом канале со сглаженным поворотом, в канале с резким поворотом и в канале с преградой. Характеристики исследованных течений схожи с характеристиками течения в неглубокой равнинной реке. В работе [10] также приведены результаты исследования влияния на течение и образование ледовых заторов размера ледяных частиц, взаимодействия частиц между собой, а также толщины слоя ледяных частиц.

Численное моделирование течения на участке р. Томь. С начала XXI в на реке Томь вблизи города Томска вновь начали наблюдаться ледовые заторы, которые вызывают значительный подъем уровня воды в реке и затопление территорий в пойме [14]. Таким образом, прогнозирование возможного появления зон локального затопления во время весенних паводков является актуальной задачей.

Объектом исследования в данной работе является участок р. Томь длиной 21 *км* от пригорода Томска до г. Северска. Средняя скорость течения р. Томь – 0,33 м/с. Средняя глубина реки в районе г. Томска – 2,5 м, ширина речного русла в среднем составляет 800 *м*. На исследуемом участке река имеет однорукавное русло с островами небольшой площади.

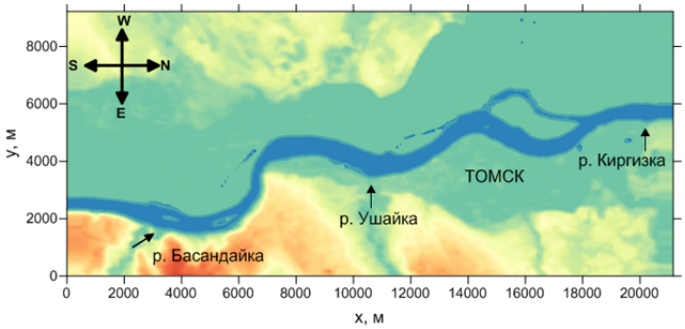


Рис. 2. Объект исследования

Как известно, во время ледохода лед с наибольшей вероятностью накапливается в поворотах русла из-за того, что течение перестраивается в таких областях, а льдины, обладающие существенной инерцией, продолжают двигаться прямолинейно. Таким образом, скопление льдин на исследуемом участке вероятно на отрезке сужения русла (между 5 и 7 км), а также за островами и на участке у левого берега между 14 и 18 км. Кроме того, обширные отмели на р. Томь у берегов и в поворотах увеличивают сопротивление русла, также способствуя накоплению как крупных, так и небольших льдин.

Моделирование вскрытия реки ото льда

В начальный момент времени предполагается, что река течет подо льдом. Дополнительное сопротивление со стороны ледяной крышки потоку задается увеличением коэффициента трения потока c_f. Предполагается, что в определенный момент после установления течения река

вскрывается внезапно на всей исследуемой области и ледяная крышка становится континуумом ледяных частиц с характерным размером 1 м.

В начальный после вскрытия реки момент времени скорость фазы частиц равна 0 и потому появляется значительное по величине сопротивление со стороны частиц потоку, вызывающее торможение потока и, соответственно, подъем свободной поверхности (рис. 3).

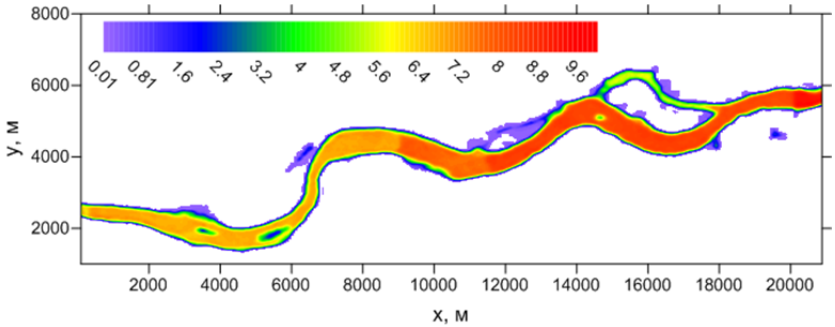


Рис. 3 Глубина воды в начальный после вскрытия момент времени

По мере того как частицы увлекаются потоком, разность скоростей фаз уменьшается, межфазное трение уменьшается и понижается и стабилизируется уровень воды.

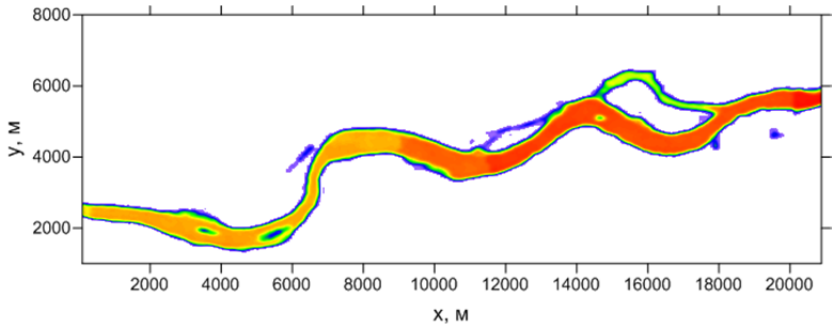


Рис. 4. Глубина воды после установления двухфазного течения

Проведены вычислительные эксперименты по исследованию затопления р. Томь прибрежных территорий в результате вскрытия реки перед началом ледохода. Анализ полученных результатов показал, что разработанная модель правильно предсказывает подтопление низин и островов,

связанное с увеличением уровня воды и увеличением сопротивления потоку со стороны ледяных частиц.

Литература

1. Бузин В.А., Зиновьев А.Т. Ледовые процессы и явления на реках и водохранилищах (обзор современного состояния проблемы). Барнаул : ИВЭП СО РАН, 2009. 167 с.
2. Beltaos S. Rivers ice jams: theory, case studies, and applications // Journal of Hydraulic Engineering. 1983. Vol. 109, No. 10. Pp. 1338–1359.
3. Берденников В.П. Динамические условия образования заторов // Труды государственного гидрологического института. 1964. № 110. С. 3–11.
4. Пушистов П.Ю., Данчев В.Н. Информационно-вычислительные комплексы водных объектов бассейна Оби. Саарбрюккен : LAP, 2013. 151 с.
5. Shen H.T., Su J., and Liu L. SPH Simulation of River Ice Dynamics // Journal of Computational Physics. 2000. Vol. 165, No. 2. Pp. 752–770.
6. Shen H.T. Mathematical modeling of river ice processes // Cold Regions Science and Technology. 2010. Vol. 62, No. 1. Pp. 3–13.
7. Wang X., Jiang Z., Zhang A., Zhou Z., and An J. Two-phase flow numerical simulation of a bend-type ice sluice in the diversion water channel of powerhouse // Cold Regions Science and Technology. 2012. Vol. 81. Pp. 36–47.
8. Wang J., Chen P. Progress in studies on ice accumulation in river bends // Journal of hydrodynamics. 2011. Vol. 23, No. 6. Pp. 737–744.
9. Нигматулин Р.И. Динамика многофазных сред. М. : Наука, 1987. Ч. 1. 464 с.
10. Чуруксаева В.В., Старченко А.В. Численное исследование двухфазного течения жидкости с легкими частицами в открытых каналах // Вестник Томского государственного университета. Математики и механика. 2016. № 6 (44). С. 88–103.
11. Бубенчиков А.М., Старченко А.В. Численные модели динамики и горения аэродисперсных смесей в каналах. Томск : Изд-во Том. ун-та, 1998. 236 с.
12. Launder B.E., Spalding D.B. The numerical computation of turbulent flows // Computer Methods in Applied Mechanics and Engineering. 1974. Vol. 2, No. 3. Pp. 269–289.
13. Gidaspov D. Multiphase Flow and Fluidization: Continuum and Kinetic Theory Descriptions. Boston : Academic Press, 1994. 457 p.
14. Земцов В.А., Вершинин Д.А., Инишев Н.Г. Имитационное моделирование заторов (на примере р. Томь, Западная Сибирь) // Лед и Снег. 2014. № 3 (127). С. 59–68.

DOI: 10.17223/9785946216531/10

Численное моделирование турбулентного течения в уличном каньоне при смешанной конвекции*

А.В. Старченко[1, 2], Е.А. Данилкин[1], А.А. Семёнова[1], Д.В. Лещинский[1]

[1] Томский государственный университет, Томск
[2] Институт мониторинга климатических и экологических систем СО РАН, Томск
starch@math.tsu.ru

Представлены математическая модель, численный метод и некоторые результаты расчета неизотермического турбулентного течения и переноса примеси в 3D уличном каньоне в условиях заметного влияния естественной конвекции.

Ключевые слова: течения с плавучестью, турбулентность, перенос примеси, уличный каньон.

Наиболее важным элементом городских территорий любого мегаполиса являются зоны плотной жилой застройки центральной части города, где требуется обеспечить в максимальной степени социальный и экологический комфорт населения при рациональном использовании ресурсов и городских земель. Основными источниками загрязнения воздуха жилых территорий являются промышленные предприятия, отопительные котельные и автомобильный транспорт. Среди них наиболее значительную долю загрязнения атмосферного воздуха в пределах жилых территорий вносит автотранспорт.

Современные методы вычислительной гидродинамики позволяют решать задачи, связанные с переносом атмосферных примесей внутри городской застройки [1]. Однако, необходимо учитывать тот факт, что правильное предсказание турбулентной структуры воздушного потока обеспечивает корректное моделирование не только полей скорости и давления, но и процессов, связанных с рассеянием и турбулентным перемешиванием выбросов в атмосфере города. Поэтому особое внимание необходимо уделять методике моделирования турбулентности, а также схеме турбулентного замыкания в рамках выбранного подхода. Следует также

* Работа выполнена при финансовой поддержке РФФИ и Администрации Томской области, грант № 16-41-700178 р_а.

отметить, что в теплое время года существенной становится неоднородность температур в уличных каньонах вследствие неравномерности нагрева ограничивающих течение поверхностей, приводящей к образованию интенсивных вихревых течений [2].

Целью данной работы является разработка в рамках подходов Рейнольдса и Буссинеска математической модели пространственных стационарных турбулентных неизотермических движений воздуха в уличных каньонах для исследования влияния характера течения на перенос примеси в условиях существенного влияния сил плавучести, в том числе и на турбулентную структуру потока.

С учетом выбранных подходов (описания турбулентного течения на основе осредненных по Рейнольдсу уравнений Навье-Стокса для несжимаемой жидкости, применения приближений Буссинеска для замыкания уравнений Рейнольдса и представления силы плавучести) система уравнений гидродинамики, теплообмена и переноса примеси будет иметь вид:

$$\frac{\partial \overline{u}_i}{\partial x_i} = 0; \tag{1}$$

$$\frac{\partial \overline{u}_i \overline{u}_j}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\nu \frac{\partial \overline{u}_i}{\partial x_j}\right) - \frac{\partial \overline{u_i' u_j'}}{\partial x_j} - \frac{g_i(\overline{T} - T_0)}{T_0}; \tag{2}$$

$$\frac{\partial \overline{T}\overline{u}_j}{\partial x_j} = \frac{\partial}{\partial x_j}\left(a \frac{\partial \overline{T}}{\partial x_j}\right) - \frac{\partial \overline{T' u_j'}}{\partial x_j}; \tag{3}$$

$$\frac{\partial \overline{c}\overline{u}_j}{\partial x_j} = \frac{\partial}{\partial x_j}\left(D \frac{\partial \overline{c}}{\partial x_j}\right) - \frac{\partial \overline{c' u_j'}}{\partial x_j} + S_c, i = 1,2,3. \tag{4}$$

Здесь $\overline{u}_i, u_i'$ – осредненная и пульсационная компоненты (i=1,2,3) вектора скорости, T – температура, p – давление, $g_i(0,0,-g)$ – компоненты ускорения свободного падения, с – концентрация примеси, S_c – интенсивность поступления примеси, ρ, ν, a, D – плотность, кинематическая вязкость, температуропроводность и коэффициент диффузии.

Для замыкания уравнений (2)–(4) используется градиентная гипотеза Буссинеска:

$$\overline{u_i'u_j'} = -\nu_t\left(\frac{\partial \overline{u}_i}{\partial x_j}+\frac{\partial \overline{u}_j}{\partial x_i}\right)+\frac{2}{3}\delta_{ij}k; i,j=1,2,3;$$

$$\overline{T'u_j'} = -\frac{\nu_t}{\mathrm{Pr}_t}\frac{\partial \overline{T}}{\partial x_j};\quad \overline{c'u_j'} = -\frac{\nu_t}{Sc_t}\frac{\partial \overline{c}}{\partial x_j}; j=1,2,3.$$

Для вычисления турбулентной вязкости ν_t и кинетической энергии турбулентности k применялась двухпараметрическая «k-ε» модель турбулентности:

$$\frac{\partial k\overline{u}_j}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\left(\nu+\frac{\nu_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right)+P+G-\varepsilon;\sigma_k=1.0; \qquad (5)$$

$$\frac{\partial \varepsilon\overline{u}_j}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\left(\nu+\frac{\nu_T}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right)+C_{\varepsilon 1}\frac{\varepsilon}{k}\left(P+C_{\varepsilon 3}G\right)-C_{\varepsilon 2}\frac{\varepsilon^2}{k}; \qquad (6)$$

$$C_{\varepsilon 1}=1.44, C_{\varepsilon 2}=\tanh(|\overline{u}_3|/\sqrt{\overline{u}_1^2+\overline{u}_2^2});$$
$$C_{\varepsilon 3}=1.92, \sigma_\varepsilon=1.22;$$

$$\nu_t = C_\mu k^2/\varepsilon; C_\mu=0.09; \mathrm{Pr}_t=0.9; Sc_t=0.5; \qquad (7)$$

$$P=\nu_t\left(\frac{\partial \overline{u}_i}{\partial x_j}+\frac{\partial \overline{u}_j}{\partial x_i}\right)\frac{\partial \overline{u}_i}{\partial x_j}; \qquad G=\delta_{i3}\frac{g_i}{T_0}\frac{\nu_t}{\mathrm{Pr}_t}\frac{\partial \overline{T}}{\partial x_i}. \qquad (8)$$

Для задания значений турбулентных параметров вблизи твёрдой поверхности, а также трения и теплового потока на стенках при известной температуре ограждающих поток поверхностей используется метод пристенных функций Ченга-Лаундера. Предполагается отсутствие осаждения примеси на стенках, поэтому для концентрации задавались нулевые значения производной по нормали к поверхности. На входе в расчетную область значения параметров задачи считались известными. На выходной границе

использовались простые градиентные условия – равенство нулю производной по нормали для зависимых переменных задачи. На верхней границе, расположенной на достаточной высоте над поверхностными препятствиями, а также на боковых плоскостях для достаточно протяженного по ширине уличного каньона применялись простые градиентные условия.

Решение рассматриваемой задачи (1)–(8) выполнялось численно с помощью метода конечного объема. Расчётная область составляется фиксированным числом непересекающихся конечных объёмов так, чтобы каждый узел расчётной сетки содержался в одном объёме [1]. Значения компонент скорости ветра определяются на гранях конечных объёмов, а скалярные характеристики - в центре. Разбив таким образом расчётную область, дифференциальные уравнения интегрируются по каждому конечному объёму [3]. Для вычисления интегралов используются кусочно-линейные профили, которые описывают изменение переменной между узлами. Аппроксимация конвективных членов уравнений переноса выполняется с использованием схемы MLU Ван Лира [1], а дискретизация диффузионных слагаемых осуществляется с помощью центральных разностей. В результате такого интегрирования получается неявный дискретный аналог дифференциальных уравнений, в который входят значения переменной в нескольких соседних узлах. На основе процедуры SIMPLE [3] осуществляется согласование полей скорости и давления. Системы сеточных уравнений решаются явным методом Булеева. При расчёте течений в областях сложной геометрии использовался метод фиктивных областей, суть которого заключается в том, что используется обычная структурированная сетка, значения векторных и скалярных величин в области преграды не изменяются, а на границах фиктивных конечных объемов равны нулю адвективные потоки.

Предложенная математическая модель была применена для исследования аэродинамики потока и переноса примеси, поступающей от точечного источника, в уличном каньоне. Область исследования представлена на рис. 1.

Скорость входного потока составляла 5 м/с, уровень интенсивности турбулентности не превышал 5%. Расчеты проводились на сетке 102x63x92. Источник примеси постоянной интенсивности располагался вблизи поверхности в середине основания каньона на высоте 0,125 м над поверхностью. На рис. 2 представлены линии тока и черно-белые карты логарифма нормированной концентрации примеси в вертикальной плоскости, проходящей через середину каньона (x_2 =15 м).

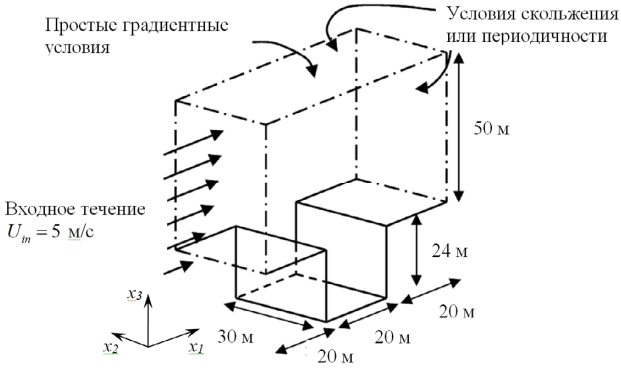


Рис. 1. Расчетная область

Из рисунка видно, что в каньоне образуется устойчивое закрученное движение с застойной областью в центре каньона и с более интенсивным движением в пристенной области. Такой характер аэродинамики потока способствует переносу примеси к подветренной стенке каньона с частичным ее выносом в надканьонное пространство и возвращением части примеси в зону вращательного движения [1].

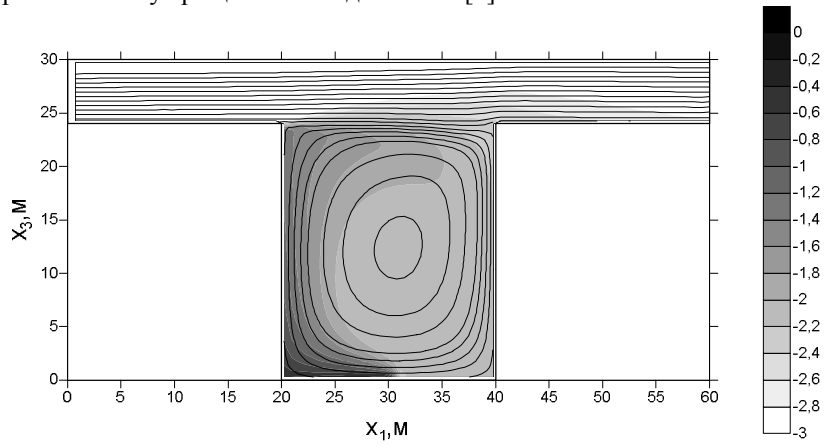


Рис. 2. Линии тока и черно-белые карты log10(c/cmax) в сечении x_2 =15 м

На рис. 3 представлены результаты сравнения рассчитанных нормированных значений концентрации примеси вблизи наветренной и подветрен-

ной вертикальных стенок каньона с измерениями Хойдыша и Дэбберта, выполненными в ветровом туннеле [4]. Из рисунка видно, что имеет место удовлетворительное согласование расчетов с экспериментальными данными. В то же время следует отметить, что представленные в работах [1, 2] результаты сопоставления расчетов численного моделирования с применением вихреразрешающих моделей турбулентности демонстрируют подобный характер согласования с данными измерений Хойдыша и Дэбберта [4].

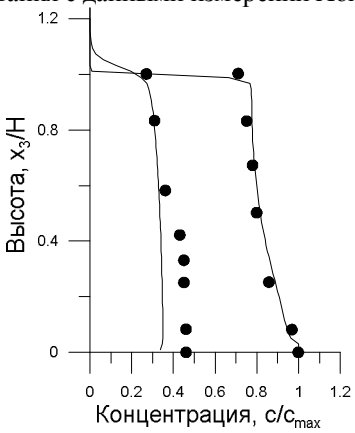


Рис. 3. Рассчитанные и измеренные [4] вертикальные профили нормированной концентрации примеси у наветренной (слева) и подветренной (справа) сторон каньона

Также в работе было произведено исследование влияние нагрева ограждающих поток поверхностей на аэродинамику и перенос примеси в каньоне. Рассматривались три варианта:

А) температура наветренной стенки была выше температуры окружающего воздуха на 10 градусов;

Б) температура нижней стенки была выше температуры окружающего воздуха на 10 градусов;

В) температура подветренной стенки была выше температуры окружающего воздуха на 10 градусов.

Во всех рассмотренных вариантах температура окружающего воздуха была равна 300 К, а скорость набегающего потока 1,18 м/с.

В случае нагретой наветренной стороны уличного каньона, происходит образование восходящего потока воздуха у наветренной стороны. При этом, движение образовавшегося восходящего потока направленно в

сторону, противоположную движению основного вихря. Это приводит к поджатию основного вихря и образованию нового в зоне между основным вихрем и наветренной стеной здания. На рис. 4 представлены линии тока и распределение логарифма нормированной концентрации примеси, соответствующие рассматриваемому случаю (А).

В случае, когда нагрето дно уличного каньона (вариант Б), образующийся восходящий поток воздуха усиливает основной вихрь и увеличивает среднюю скорость кругового движения, что способствует более интенсивному выветриванию примеси из уличного каньона (рис. 5).

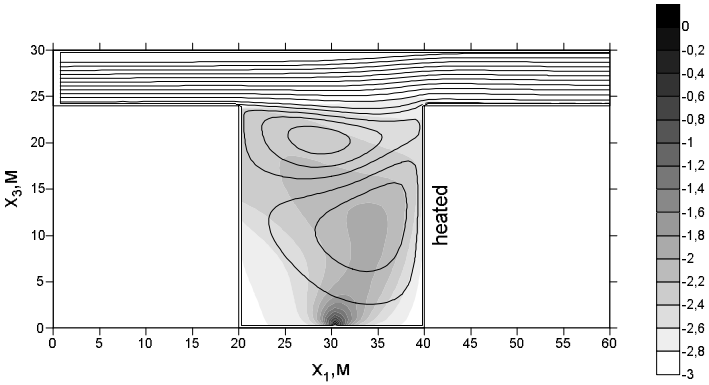


Рис. 4. Линии тока и черно-белые карты log10(c/cmax) в сечении x_2 =15 м, соответствующие варианту А

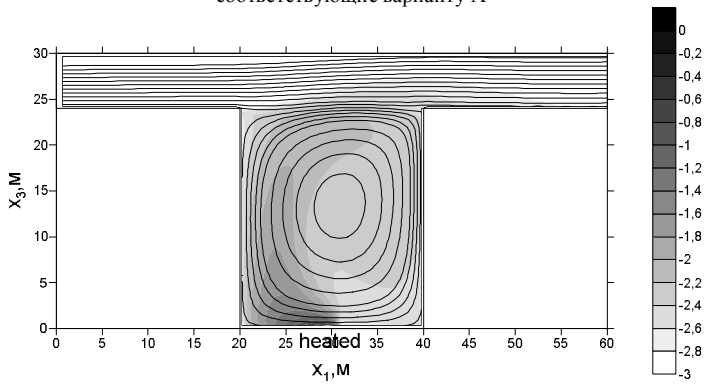


Рис. 5. Линии тока и черно-белые карты log10(c/cmax) в сечении x_2 =15 м, соответствующие варианту Б

В случае, когда нагрета подветренная сторона (вариант В), восходящий поток вблизи нагретой подветренной стены здания также усиливает основной вихрь и увеличивает среднюю скорость кругового движения, вследствие чего границы основного вихря расширяются выше уровня крыш уличного каньона (рис. 6). Полученные результаты хорошо согласуются с результатами математического моделирования, полученными в работах [1, 2].

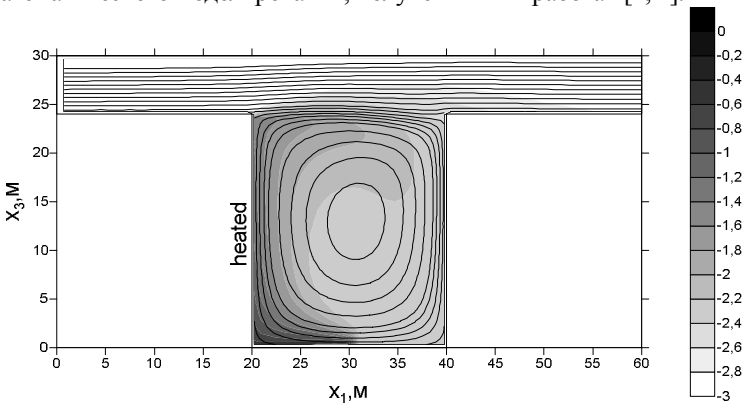


Рис. 6. Линии тока и черно-белые карты log10(c/cmax)
в сечении x_2 =15 м, соответствующие варианту В

Таки образом, для исследования структуры течения и процессов переноса примеси в уличных каньонах разработана трехмерная модель турбулентного движения, учитывающая влияние температурной неоднородности. На основе разработанной модели проведено исследование влияние температурной неоднородности на структуру течения и уровень загрязнения в уличном каньоне. Верификация разработанной модели проведена на задаче моделирования турбулентного течения и переноса примеси в ветровом туннеле [4].

Литература

1. Старченко А.В., Нутерман Р.Б., Данилкин Е.А. Численное моделирование турбулентных течений и переноса примеси в уличных каньонах. Томск : Изд-во Том. гос. ун-та, 2015. 252 с.
2. Wang P. Zhao D., Wang W., Mu H., Cai G., Liao C. Thermal Effect on Pollutant Dispersion in an Urban Street Canyon // Int. J. Environ. Res. 2011. Vol. 5, № 3. Pp. 813–820.
3. Патанкар С. Численные методы решения задач теплообмена и динамики жидкости / Пер. с англ. М. : Энергоатомиздат, 1984. 149 с.
4. Hoydysh W.G., Dabberdt W.F. Kinematics and dispersion characteristics of flows in asymmetric street canyons // Atmospheric Environment. 1988. Vol. 22. P. 2677–2689.

DOI: 10.17223/9785946216531/11

Исследование математической модели «хищник–жертва» с учётом внутривидовой конкуренции и ареала обитания в двумерном приближении

А.Р. Чу, М.Д. Михайлов

Томский государственный университет, Томск
chu.antoniy@gmail.com

В основе большинства различных математических моделей, описывающих динамику численности популяции и взаимодействующих по принципу «хищник-жертва», лежит классическая модель Лотки-Вольтерра. В данной работе используется модель, приведенная к более простому виду и с меньшим числом коэффициентов [1], подобранных с учётом устойчивости математической модели. В модели учитываются условия внутривидовой конкуренции среди хищников и жертв и ареал обитания в двумерном приближении. Реализация данной пространственной модели осуществляется с помощью неявного численного метода. Даётся сравнение результатов численного счёта с результатами из [2], из которого следует, что процессы, описываемые для двумерного случая, также являются устойчивыми.
Ключевые слова: математическая модель, аппроксимация, устойчивость, узел, фокус, хищник-жертва, ареал обитания, внутривидовая конкуренция.

Модель Базыкина [1], представляющая модификацию классической модели Лотки-Вольтерра, имеет следующий вид:

$$\begin{cases} \dfrac{du}{dt} = u - uv, \\ \dfrac{dv}{dt} = -\gamma u + uv, \end{cases} \tag{1}$$

с соответствующими начальными условиями

$$\begin{cases} u(0) = u_0, \\ v(0) = v_0. \end{cases}$$

С учётом внутривидовой конкуренции среди хищников и жертв система (1) приводится к следующему виду:

$$\begin{cases} \dfrac{du}{dt} = u - uv - c_1 u^2, \\ \dfrac{dv}{dt} = -\gamma u + uv - c_2 v^2, \\ u(0) = u_0, \\ v(0) = v_0. \end{cases} \qquad (2)$$

Исследование устойчивости системы (2), а также условия, обеспечивающие устойчивость процессов, описываемых системой (2), приведены в [2] (см. неравенства (10) и (11)). Для определения параметров c_1, c_2, γ фиксируется два параметра c_1, γ и находится промежуток изменений для c_2. Часть полученных результатов приведена в таблицах 1 и 2.

Таблица 1

Случай устойчивого фокуса

γ \ c_1	0,01	0,31	0,61	0,91
0,01	$0<c_2<1.58$	$0<c_2<1.49$	$0<c_2<1.40$	$0<c_2<1.33$
0,91	$0.18<c_2<1.91$	$0<c_2<1.57$	$0<c_2<1.21$	$0<c_2<0.74$
1,81	$0.54<c_2<2.16$	$0.038<c_2<1.45$	False	False
2,71	$0.88<c_2<2.38$	$0<c_2<1.02$	False	False

Таблица 2

Случай устойчивого узла

γ \ c_1	0,01	0,31	0,61	0,91
0,01	$c_2>1.58$	$c_2>1.49$	$c_2>1.40$	$c_2>1.33$
0,91	$0<c_2<0.18 \| c_2>1.91$	$c_2>1.57$	$c_2>1.21$	$c_2>0.74$
1,81	$0<c_2<0.54 \| c_2>2.16$	$0<c_2<0.038 \| c_2>1.45$	False	False
2,71	$0<c_2<0.88 \| c_2>2.38$	$c_2>1.02$	False	False

Рассматривается модификация модели Базыкина-Свирежева с учётом внутривидовой конкуренции и ареала обитания в двумерном приближении. Ищутся функции

$u(x_1, x_2, t), v(x_1, x_2, t) \in C_1^{2,2}(G) \cap C(\overline{G}), \overline{G} = \{(x_1, x_2, t) \mid 0 \le x_1 \le L_1; 0 \le x_2 \le L_2; 0 \le t \le T\}$,
удовлетворяющие системе уравнений:

$$\begin{cases} \dfrac{\partial u}{\partial t} = D_{1,u} \dfrac{\partial^2 u}{\partial x_1^2} + D_{2,u} \dfrac{\partial^2 u}{\partial x_2^2} + u - uv - c_1 u^2, \\ \dfrac{\partial v}{\partial t} = D_{1,v} \dfrac{\partial^2 v}{\partial x_1^2} + D_{2,v} \dfrac{\partial^2 v}{\partial x_2^2} - \gamma u + uv - c_2 v^2 \end{cases} \tag{3}$$

с соответствующими начальными:

$$\begin{cases} u(x_1, x_2, t) = \varphi_1(x_1, x_2, t), \\ v(x_1, x_2, t) = \varphi_2(x_1, x_2, t) \end{cases} \tag{4}$$

и граничными условиями:

$$\left.\frac{\partial u}{\partial x_1}\right|_{\substack{x_1=0,\\ x_1=L_1}} = \left.\frac{\partial v}{\partial x_1}\right|_{\substack{x_1=0,\\ x_1=L_1}} = \left.\frac{\partial u}{\partial x_2}\right|_{\substack{x_2=0,\\ x_2=L_2}} = \left.\frac{\partial v}{\partial x_2}\right|_{\substack{x_2=0,\\ x_2=L_2}} = 0. \tag{5}$$

Начальные функции $\varphi_1(x)$ и $\varphi_2(x)$ задаются в виде, представленном на рисунках 1 и 2.

Для решения задачи (3)–(5) используется неявный разностный метод. Для этого область $\overline{G}$ покрывается равномерной сеткой вида

$$\overline{\omega}_{h,\tau} = \overline{\omega}_{h_1} \times \overline{\omega}_{h_2} \times \overline{\omega}_\tau; \overline{\omega}_{h_1} = \left\{ x_{1,j} \,\middle|\, x_{1,j} = j \cdot h, j = \overline{0, N_1}, h_1 = \frac{L_1}{N_1} \right\};$$

$$\overline{\omega}_{h_1} = \left\{ x_{2,k} \,\middle|\, x_{2,k} = k \cdot h, k = \overline{0, N_2}, h_2 = \frac{L_2}{N_2} \right\}; \overline{\omega}_\tau = \left\{ t_n \,\middle|\, t_n = n \cdot \tau, n = \overline{0, M} \right\},$$

и дифференциальная задача (3)–(5) аппроксимируется соответствующей разностной с использованием неявной схемы. В результате будем иметь:

$$\begin{cases} \dfrac{u_{j,k}^{n+1} - u_{j,k}^n}{\tau} = D_{1,u} \dfrac{u_{j+1,k}^{n+1} - 2u_{j,k}^{n+1} + u_{j-1,k}^{n+1}}{h^2} + \\ + D_{2,u} \dfrac{u_{j,k+1}^n - 2u_{j,k}^n + u_{j,k-1}^n}{h^2} + u_{j,k}^{n+1}(1 - v_{j,k}^n - c_1 u_{j,k}^n), \\ \dfrac{v_{j,k}^{n+1} - v_{j,k}^n}{\tau} = D_{1,v} \dfrac{v_{j+1,k}^{n+1} - 2v_{j,k}^{n+1} + v_{j-1,k}^{n+1}}{h^2} + \\ + D_{2,v} \dfrac{v_{j,k+1}^n - 2v_{j,k}^n + v_{j,k-1}^n}{h^2} + v_{j,k}^{n+1}(-\gamma + u_{j,k}^n - c_2 v_{j,k}^n). \end{cases} \tag{6}$$

За начальные условия берем проекции функций $\varphi_1(x)$ и $\varphi_2(x)$ на равномерную сетку $\overline{\omega}_{h,\tau}$, приведённые в таблицах 3 и 4.

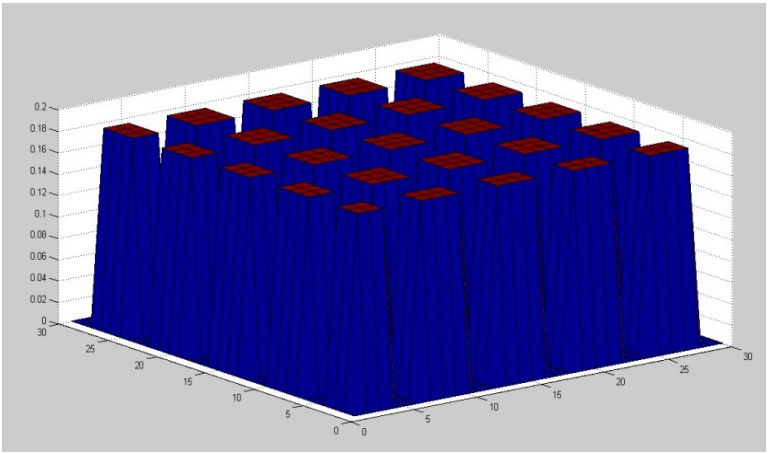


Рис. 1. Плотность популяции жертв в начальный момент времени

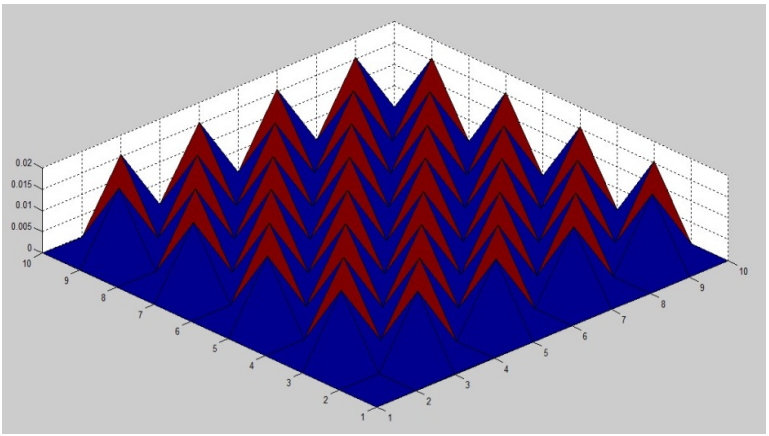


Рис. 2. Плотность популяции хищников в начальный момент времени

Таблица 3

Проекция функции $\varphi_1(x)$

	1	2	3	4	5	6	7	8	9	10	11	12
1	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0.1837	0.1837	0.1837	0	0	0.1837	0.1837	0.1837	0.1837	0	0
3	0	0.1837	0.1837	0.1837	0	0	0.1837	0.1837	0.1837	0.1837	0	0
4	0	0.1837	0.1837	0.1837	0	0	0.1837	0.1837	0.1837	0.1837	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0.1837	0.1837	0.1837	0	0	0.1837	0.1837	0.1837	0.1837	0	0
8	0	0.1837	0.1837	0.1837	0	0	0.1837	0.1837	0.1837	0.1837	0	0
9	0	0.1837	0.1837	0.1837	0	0	0.1837	0.1837	0.1837	0.1837	0	0
10	0	0.1837	0.1837	0.1837	0	0	0.1837	0.1837	0.1837	0.1837	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0.1837	0.1837	0.1837	0	0	0.1837	0.1837	0.1837	0.1837	0	0
14	0	0.1837	0.1837	0.1837	0	0	0.1837	0.1837	0.1837	0.1837	0	0
15	0	0.1837	0.1837	0.1837	0	0	0.1837	0.1837	0.1837	0.1837	0	0
16	0	0.1837	0.1837	0.1837	0	0	0.1837	0.1837	0.1837	0.1837	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0

Таблица 4

Проекция функции $\varphi_1(x)$

	1	2	3	4	5	6	7	8	9	10	11	12
1	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0
3	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0
4	0	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0
5	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0
6	0	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0
7	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0
8	0	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0
9	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0
10	0	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0
11	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0
12	0	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0
13	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0
14	0	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0
15	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0
16	0	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0
17	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0.0155	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0

Граничные условия задаются следующим образом:

$$u_{0,k}^{n+1} = u_{1,k}^{n+1}; u_{N_1,k}^{n+1} = u_{N_1-1,k}^{n+1}; v_{0,k}^{n+1} = v_{1,k}^{n+1}; v_{N_1,k}^{n+1} = v_{N_1-1,k}^{n+1}; k = \overline{0,N_2}; n = \overline{0,M-1}; \quad (7)$$

$$u_{j,0}^{n+1} = u_{j,1}^{n+1}; u_{j,N_2}^{n+1} = u_{j,N_2-1}^{n+1}; v_{j,0}^{n+1} = v_{j,1}^{n+1}; v_{j,N_2}^{n+1} = v_{j,N_2-1}^{n+1}; j = \overline{0,N_1}; n = \overline{0,M-1}.$$

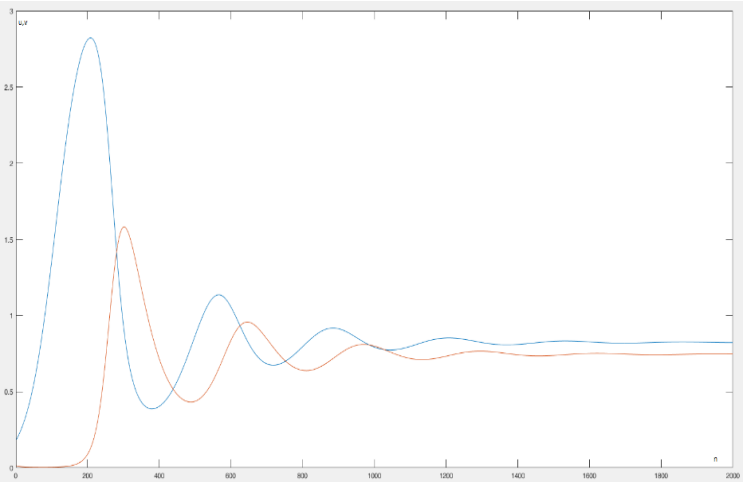


Рис. 3. Изменение плотности популяций жертв и хищников в узле сетки (8;8) с течением времени (случай устойчивого фокуса)

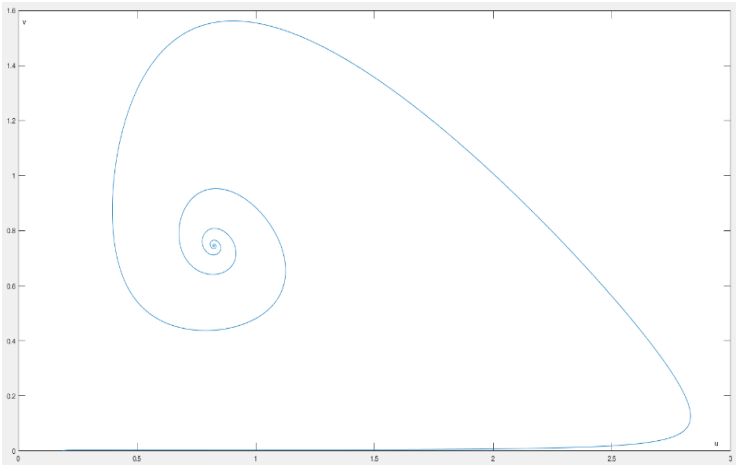


Рис. 4. Изменение плотности популяций жертв и хищников в узле сетки (8;8) в фазовой плоскости (случай устойчивого фокуса)

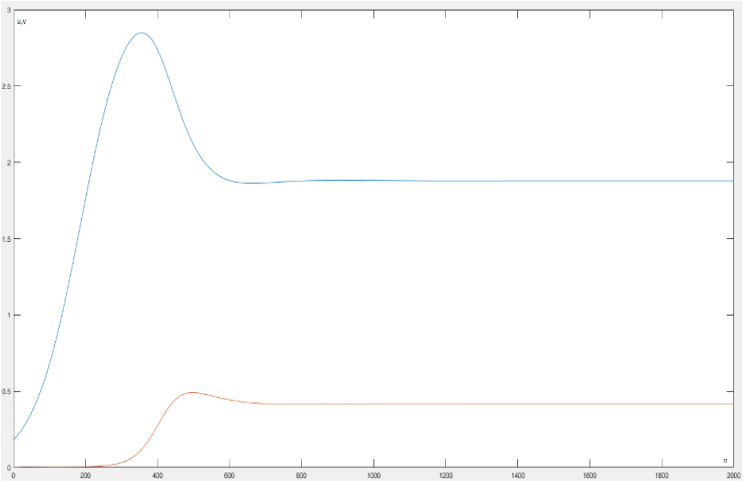


Рис. 5. Изменение плотности популяций жертв и хищников в узле сетки (8;8) с течением времени (случай устойчивого узла)

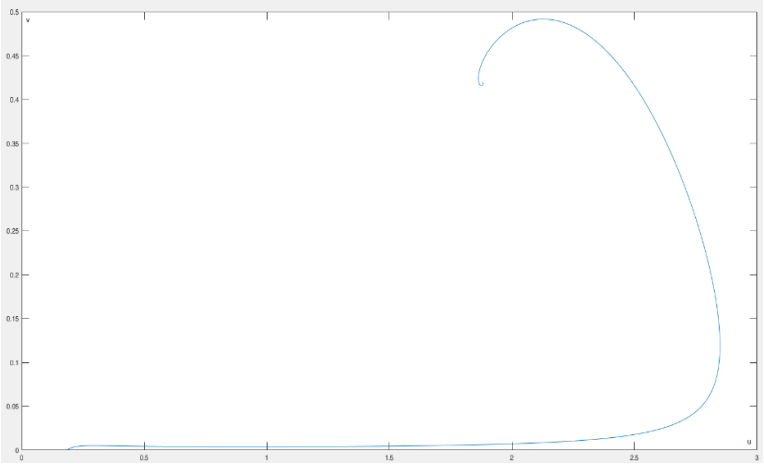


Рис. 6. Изменение плотности популяций жертв и хищников в узле сетки (8;8) в фазовой плоскости (случай устойчивого узла)

Исследуются вопросы аппроксимации, устойчивости и сходимости численного метода (6)–(7). Показано, что разностная задача аппроксимирует соответствующую дифференциальную (3)–(5) с первым порядком по τ и h и абсолютно устойчива по начальным данным. Отсюда, по теореме Лакса [3] вытекает, что решение разностной задачи сходится к решению дифференциальной задачи.

Для реализации численного метода используются значения параметров переменных, обеспечивающих устойчивость системы (3)–(5), взятые из таблиц 1 и 2, приведённых ранее:

для устойчивого фокуса

$$D_{1,u} = 0.1; D_{2,u} = 0.1; D_{1,v} = 10; D_{2,v} = 10; c_1 = 0.31; c_2 = 0.05; \gamma = 0.91;$$

для устойчивого узла

$$D_{1,u} = 0.1; D_{2,u} = 0.1; D_{1,v} = 10; D_{2,v} = 10; c_1 = 0.31; c_2 = 1{,}61; \gamma = 0.91.$$

На рис. 3–6 приведены результаты численных расчётов двумерной задачи в узле (8;8) пространственной сетки для сравнения с результатами, полученными для точечной и одномерной модели соответственно. Сравнение показало качественное совпадение результатов.

В заключении следует сказать, что определение оптимальных параметров системы «хищник-жертва» в двумерном приближении по предложенной в работе методике оправдало себя и может быть рекомендовано в усложнённых вариантах задач такого рода. Например, когда учитывается возраст особей популяции, половая структура и другие факторы.

Литература

1. Базыкин А.Д. Нелинейная динамика взаимодействующих популяций. Москва-Ижевск : Институт компьютерных технологий, 2003. С. 225–238.
2. Чу А.Р., Михайлов М.Д. Исследование математической модели «хищник–жертва» с учётом внутривидовой конкуренции // Всероссийская молодежная научная конференция «Все грани математики и механики» : сборник статей / под ред. А.В. Старченко. Томск : Издательский Дом Томского государственного университета, 2016. С. 118–124.
3. Меркулова Н.Н., Михайлов М.Д. Методы приближённых вычислений / под ред. А.В. Старченко. Томск : Издательский Дом Томского государственного университета, 2014. 762 с.

DOI: 10.17223/9785946216531/12

Программный комплекс для построения распределенной вычислительной системы со слабо связанными совокупностями вычислителей

В.А. Крутиков[1], И.А. Ботыгин[1,2], В.С. Шерстнёв[2]

[1] Институт мониторинга климатических и экологических систем СО РАН, Томск
[2] Томский политехнический университет, Томск
krutikov@imces.ru, bia@tpu.ru, vss@tpu.ru

Представлен комплекс программных средств для построения распределенной информационно-вычислительной платформы со слабо связанными совокупностями вычислителей. Технологическая схема реализации и развертывания платформы включает кластер узлов обработки и кластер узлов хранения, предоставляющих свои вычислительные мощности для выполнения задач исследователя по запросу. В каждом из кластеров выделен главный узел – командный центр. Центр управления хранением предоставляет программный интерфейс для размещения данных и доступа к ним. Центр управления вычислениями координирует функциональную обработку данных, руководствуясь инструкциями, полученными от исследователей. Функциональные приложения реализованы в форме jnlp-файлов, что обеспечивает им свободный доступ к объектам и взаимодействие в распределенной структуре платформы.
Ключевые слова: распределенные вычисления, grid-технологии, вычислительный кластер.

Достижение требуемой производительности в вычислениях, как правило, обеспечивается использованием многопроцессорных вычислительных систем с применением параллельных вычислений [1, 2]. Основная цель параллельных вычислений – уменьшение времени решения задачи. Для решения трудоемких задач такие вычисления могут быть запущены, например, на суперкомпьютерах или вычислительных кластерах. Помимо аппаратных многопроцессорных вычислительных ресурсов (кластеров), существует и большое множество распределенных вычислительных систем, объединяющих в единый комплекс территориально разнесенные компьютеры – grid-системы [3–14].

В настоящей работе представлены результаты проектирования и разработки программного обеспечения динамически компонуемой платформно-независимой инфраструктуры распределенной гетерогенной вычислительной среды (grid-системы) для реализации сложных вычислительно емких процессов при исследовании и прогнозе природно-климатических изменений.

В функциональной структуре и технологической схеме взаимодействия основных компонентов информационно-вычислительной платформы выделены и спроектированы менеджер инфраструктуры, сетевой менеджер, менеджер вычислений, менеджер хранилища, менеджер авторизации, web-менеджер, менеджер планирования загрузки (рис. 1).

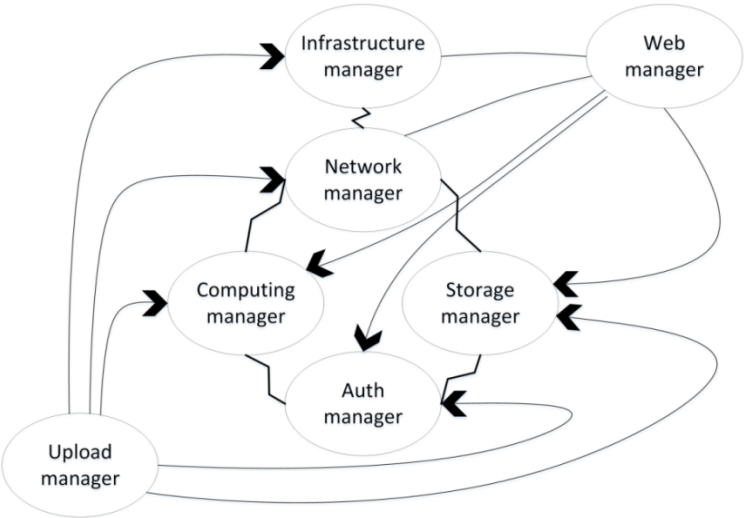


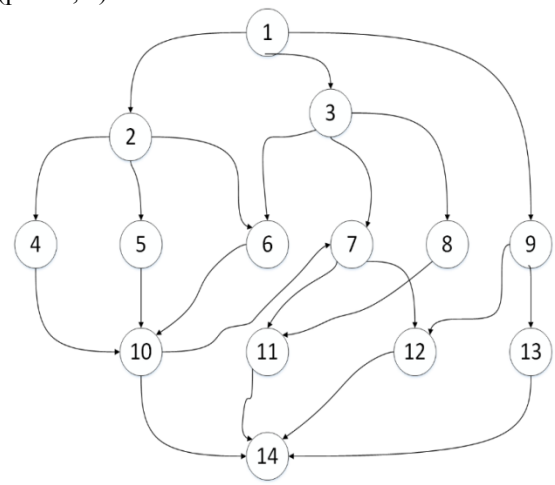
Рис. 1. Схема взаимодействия основных компонентов платформы

Основой для распределенной системы обработки данных служит модель MapReduce. Топологии вычисления задается исследователем в виде графа. Интерфейс среды позволяет исследователю сформировать свой управляющий граф последовательно-параллельных вычислений с необходимым количеством уровней распределения и синхронизации вычислений, а также выбрать необходимое количество вычислительных ресурсов для реализации. Узлы кластера вычислений получают из центра управления программные модули обработки данных и сформированные в

соответствии с задачей буферы входных данных. Специальный компонент центра вычислений определяет порядок выполнения вычислительных модулей и процесс их распределения по физическим узлам среды, осуществляет контроль параллельного выполнения и финальный сбор выходных данных.

В процессе подготовки задания к вычислению с помощью грид-системы исследователь формирует карту задач в виде графа, на которой указываются задачи и их зависимости между собой. Точнее, указываются связи тех задач, которым необходимо дождаться выполнения предыдущего этапа вычислений (рис. 2, *б*).

Файловые потоки данных
1: F1-2, F1-3, F1-9;
2: F2-4, F2-5, F2-6;
3: F3-6, F3-7, F3-8;
4: F4-10;
5: F5-10;
6: F6-10;
7: F7-11, F7-12;
8: F8-11;
9: F9-12, F-13;
10: F10-7; F10-14;
11: F11-14;
12: F12 -14;
13: F13-14.



а *б*

Рис. 2. Пример графа вычислений и потоков данных

В процессе формирования карты-графа исследователю необходимо сформировать для каждого узла функциональный модуль вычислений (исполняемый программный код), который должен быть выполнен, а также набор данных, с которыми будут производиться вычисления. Заметим, что каждая вершина карты-графа формирует именные файловые потоки данных, необходимые для работы последующих модулей вычислений (рис. 2, *а*).

Центр управления в каждый момент времени осуществляет мониторинг доступности и занятости каждого узла. Если в очереди задач есть

задача, нуждающаяся в обработке, центр назначает свободному узлу ее выполнение (рис. 3). Назначение задачи для выполнения происходит путем направления на свободный узел вычислительного кластера исполняемого программного кода и необходимого блока данных для вычислений. После успешного выполнения задачи узел отправляет результаты центру управления.

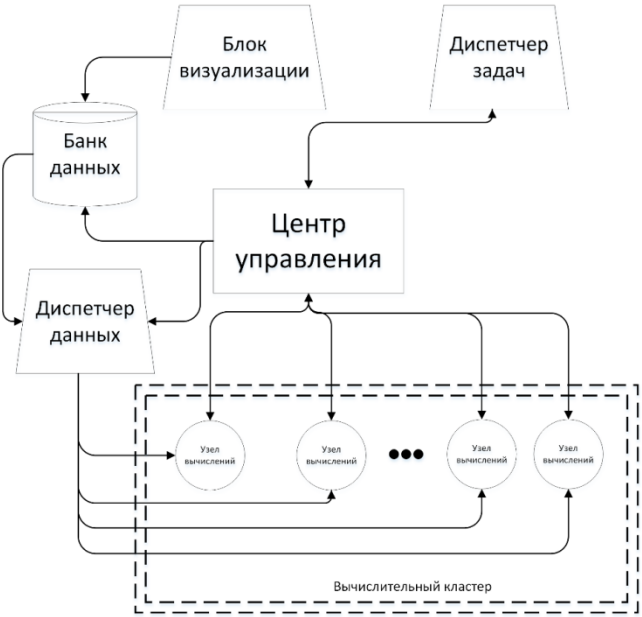


Рис. 3. Схема взаимодействия компонентов

За формирование очереди задач и контроль ее соблюдения отвечает специальный компонент программного обеспечения – диспетчер задач. Так формируется список подлежащих выполнению задач, которые могут быть выполнены в данный момент, а также очередь ожидания, куда помещаются задачи, которые не могут быть выполнены до определенных этапов.

Апробация проведенных исследований была осуществлена при разработке инфраструктуры информационно-вычислительной технологической платформы для анализа и прогноза природно-климатических процессов на основе данных многолетних замеров ультразвуковых термоан-

емометров (автоматических метеостанций). Для занесения данных были разработаны специальные средства, осуществляющие укрупнение структурной единицы, хранящейся в одном файле (формируются суточные файлы наблюдений с заданным периодом между мгновенными значениями измерений метеопараметров). Разработанные сервисы позволяют формировать метаописания к полученным файлам и оправлять их в хранилище платформы.

Реализованы три этапа первичной математической обработки всех метеорологических величин, получаемых с типовой ультразвуковой метеостанции.

Этап 1. Оценка усредненных значений метеорологических величин и других физических параметров, описывающих квазиравновесное за выбранный временной интервал состояние атмосферы в точке измерений.

Этап 2. Создание новых массивов обрабатываемых данных и вычисление на их основе физически значимых статистических моментов второго (и выше) порядка.

Этап 3. Вычисления значений параметров атмосферной турбулентности.

Таким образом, представлен комплекс программных средств для построения горизонтально-масштабируемой распределенной информационно-вычислительной платформы со слабо связанными совокупностями вычислителей (персональными компьютерами), предназначенной для информационной поддержки порталов при решении крупномасштабных научных задач, в частности, задач мониторинга, анализа и прогноза климатических процессов. Технологическая схема реализации и развертывания платформы включает в себя два функциональных компонента: кластер узлов обработки (вычислительный кластер) и кластер узлов хранения (кластер хранения), предоставляющих свои вычислительные мощности для выполнения задач исследователя по запросу. В каждом из кластеров выделен главный узел – командный центр. Основным каналом связи между компонентами платформы является сокетный интерфейс.

Полученные результаты позволяют применить наработки по управлению данными приземных метеорологических наблюдений и для оперативного сбора, систематизации и хранения данных других геофизических измерений, наблюдений в сопутствующих смежных категориях и реализовать их соответствующую математическую обработку.

Литература

1. Задачи для суперкомпьютеров, суперкомпьютерные приложения // Parallel.ru. URL: https://parallel.ru/research/challenges.html (дата обращения: 02.06.2017).

2. Параллельные вычисления при решении современных задач науки и техники. URL: http://compmech.math.msu.su/publ/Zabrodin_Parallel_comp.pdf (дата обращения: 02.06.2017).
3. Сайт Российского отделения Международной федерации грид-систем из персональных компьютеров. URL: http://desktopgrid.ru/ (дата обращения: 02.03.2017).
4. Sherstnyov V.S., Sherstnyova A.I., Botygin I.A., Kustov D.A. Distributed Information System for Processing and Storage of Meteorological Data // High Technology: Research and Applications 2015. Key Engineering Materials. Trans Tech Publications, Switzerland, 2016. Vol. 685. Pp. 867–871.
5. World Community Grid: сеть распределенных вычислений. URL: http://habrahabr.ru/company/ibm/blog/156687/ (дата обращения: 02.03.2017).
6. World Community Grid. URL: http://www.worldcommunitygrid.org/ (дата обращения: 02.03.2017).
7. Botygin I.A., Popov V.N., Tartakovsky V.A., Sherstnev V.S. Architecture of scalability file system for meteorological observation data storing // Proc. of SPIE, 21st International Symposium Atmospheric and Ocean Optics: Atmospheric Physics. 2015. Vol. 9680. Pp. 96800J-1–96800J-4.
8. Андреев А.Л, Посыпкин М.А., Заикин О.С. Гонку за терафлопсами пора заканчивать. URL: http://trv-science.ru/2015/02/10/gonku-za-teraflopsami-pora-zakanchivat/ (дата обращения: 02.03.2017).
9. Афанасьев А.П., Бычков И.В., Заикин О.С., Манзюк М.О., Посыпкин М.А., Семенов А.А. Концепция многозадачной грид-системы с гибким распределением свободных вычислительных ресурсов суперкомпьютеров. URL: http://2014.nscf.ru/TesisAll/0_NSCF_Plenar/12_189_BichkovIV.pdf (дата обращения: 02.03.2017).
10. Popov V.N., Botygin I.A., Koshelev N.V. Uniform Model of Representation Heterogeneous Data Hydrometeorological Observations // High Technology: Research and Applications 2015. Key Engineering Materials. Trans Tech Publications, Switzerland, 2016. Vol. 685. Pp. 925–929.
11. Гордов Е.П., Титов А.Г., Притупов А.А., Ботыгин И.А. Разработка веб-гис приложения для вычисления коэффициента корреляции для климатических и метеорологических характеристик // Известия Томского политехнического университета, 2014. Т. 325, № 5. С. 91–98.
12. Mishchenko P.V. Distributed computing system for solving applied tasks // 16 International conference of young specialists on micro/nanotechnologies and electron devices (EDM): [proc.]. Altai, Erlagol, 2015. Pp. 154–157.
13. Величкевич С.В., Демченко Ю.В. Использование Грид-технологий для построения распределенных САПР. URL: http://www.relarn.ru/conf/conf2005/section5/5_02.html (дата обращения: 02.03.2017).
14. Demchenko Y., van der Ham J., Yakovenko V., de Laat C., Ghijsen M., Cristea M. On-demand provisioning of Cloud and Grid based infrastructure services for collaborative projects and groups // Proceedings of the 2011 International Conference on Collaboration Technologies and Systems, 2011. Pp. 134–142.

DOI: 10.17223/9785946216531/13

Разработка и исследование алгоритма балансировки нагрузки на основе обобщенной характеристики вычислителей

Ю.В. Волков[1], И.А. Ботыгин[1,2], А.Е. Нагиев[2]

[1] Институт мониторинга климатических и экологических систем СО РАН, Томск
[2] Томский политехнический университет, Томск
yvvolkov@mail.ru, bia@tpu.ru, andrew_nagiev09@mail.ru

Описывается разработка алгоритма балансировки нагрузки на основе построения таблицы ранжирования вычислителей в зависимости от таких параметров как производительность узла, доступность, ёмкость оперативной памяти. Представлена разработка программного комплекса для проведения имитационных экспериментов по оценке эффективности разработанного алгоритма балансировки нагрузки. Имитационные задачи генерировались в виде графа вычислений с заданным количеством вершин и с различной степенью параллелизма их выполнения. Оценка эффективности осуществлялась по времени, затрачиваемому на обработку графа вычислений при различных схемах ранжирования списка вычислителей, а также в сравнении с классическими методами Round Robin и Random. Имитационное моделирование осуществлялось на компьютерах с четырех- и восьмиядерными процессорами.
Ключевые слова: балансировка нагрузки, распределенные вычисления, grid-технологии, вычислительный кластер, имитационное моделирование.

В многоядерных вычислительных структурах при параллельных вычислениях для их синхронизации особую важность приобретают алгоритмы балансировки нагрузки вычислительных узлов. Существуют различные подходы к балансировке нагрузки [1–16]. В процессе исследования были изучены и проанализированы основные существующие методы балансировки нагрузки, такие как: Random, Weighted Random, Round Robin, Weighted Round Robin, Agent-Based Adaptive, Geo-locating, Dynamic Weighted Round Robin, Data Center Aware, Token Aware, Least Connections, Weighted Least Connections, Dynamic Weighted Least Connection, Locality-Based Least Connection Scheduling, Locality-Based Least Connection Scheduling with Replication Scheduling, Least Response Time, Load-Based и др.

В результате исследования был разработан алгоритм балансировки нагрузки на основе динамического вычисления рангов вычислителей по следующим характеристикам и их комбинациям:

1) производительность процессора;

2) объем оперативной памяти вычислительного узла;

3) время доступа к вычислительному узлу;

4) производительность процессора + объем оперативной памяти вычислительного узла;

5) производительность процессора + время доступа к вычислительному узлу;

6) объем оперативной памяти вычислительного узла + время доступа к вычислительному узлу;

7) производительность процессора + объем оперативной памяти вычислительного узла + время доступа к вычислительному узлу.

Совокупность характеристик не является среднеарифметическим значением, а высчитывается с учетом того, что время доступа к узлу влияет на вычислительную мощность сильнее, чем объем оперативной памяти и производительность процессора.

Пример выбора подходящего вычислительного узла представлен на рис. 1. Для простоты каждая характеристика имеет вес от 1 (наименее производительная) до 10 (наиболее производительная).



Рис. 1. Схема балансировки нагрузки

Для оценки эффективности разработанного алгоритма балансировки была разработана имитационная среда моделирования (рис. 2).



Рис. 2. Функциональная структура среды моделирования

Среда моделирования предназначена для генерации и выполнения произвольных вычислительных задач с разной степенью параллелизма. Каждая задача состоит из определенного количества выполняемых подзадач. В качестве выполняемых подзадач выступают программные задержки, которые могут отличаться по времени между собой. Весь пользовательский интерфейс представлен в виде командной строки, в которой пользователь ведет диалог с комплексом.

Имитационная среда моделирования обладает широким спектром возможностей для создания вычислительного графа и последующей его обработки с заданием и редактированием большого количества параметров, а также с использованием различных методов балансировки нагрузки при создании вычислительного графа. Задача, создаваемая средой моделирования, представляется в виде графа вычислений, где вершины являются модулями для обработки, а связи между вершинами – связями между модулями, которые позволяют передавать данные из выхода одного модуля на вход другого. Каждый модуль графа может выполняться отдельно, однако, для этого ему необходимо подать соответствующие данные на вход, т.е. присутствует зависимость между составными частями графа.

Именно в разработанной среде моделирования на основе генерируемого графа вычислений с различным числом обрабатываемых модулей, с различной степенью параллелизма их выполнения, с генерацией времени выполнения каждого модуля на основе датчика псевдослучайных чисел проводились имитационные эксперименты по обработке различных графов вычислений с учетом различных видов балансировки нагрузки.

На рис. 3 приведен график выполнения графа вычислений из 200 модулей с одинаковым временем доступа к вычислителям и разным временем выполнения в модулях.

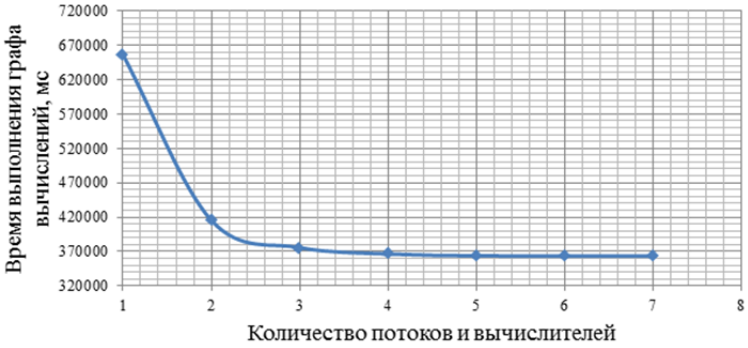


Рис. 3. Выполнение графа вычислений из 200 модулей

Компьютер, на котором была развернута среда моделирования, обладал 4 ядрами и, следовательно, способен одновременно выполнять 4 потока. Уменьшение времени при работе и на 5 и на 6 потоках выполнения связано с внутренним влиянием операционной системы (ОС) на диспетчеризацию потоков. При функционировании ОС создается несколько десятков служебных потоков и общая диспетчеризация служебными и функциональными потоками осуществляется в пользу функциональных.

Таким образом, разработан и исследован на специально созданном программном комплексе для проведения имитационных экспериментов алгоритм балансировки нагрузки в распределенных вычислительных системах. Комплекс генерирует задачу в виде графа вычислений с заданным количеством вершин (имитационных модулей вычислений) и с различной степенью параллелизма их выполнения. Для большей адекватности результатов моделирования комплекс настраивался на реальный тип процессора и реальное количество ядер в нём. При этом, для каждого

вычислительного узла (ядра) осуществлялась параметрическая настройка производительности, доступности, ёмкости оперативной памяти.

Для программно-имитационной проверки алгоритмов балансировки комплекс позволяет создавать вычислительные графы любой размерности. Временные ограничения вступают в силу только при количестве вершин в графе свыше 2000. Ограничение связано с тем, что в генерируемом вычислительном графе необходимо проверить каждую вершину на предмет наличия циклов и удалить связи, которые их образуют.

Программный комплекс написан на языке программирования Java, который отличается простотой, надежностью, повышенной отказоустойчивостью, архитектурной нейтральностью и, благодаря библиотеке java.util.concurrent, позволяет эффективно использовать возможности параллелизма при проектировании приложений.

Литература

1. Tanenbaum A.S. and Van Steen Distributed Systems: Principles and Paradigms. Upper Saddle River, NJ, USA: Pearson Education, 2007. 686 p.
2. Kshemkalyani A.D. and Singhal M. Distributed Computing: Principles, Algorithms, and Systems. Cambridge, UK : Cambridge University Press, 2008. 754 p.
3. Load Balancing Algorithms. URL: https://kemptechnologies.com/load-balancer/load-balancing-algorithms-technique (дата обращения: 20.05.2017).
4. Geo-locating load balancing. URL: https://www.google.com/patents/US20060064478 (дата обращения: 20.05.2017).
5. What is round-robin load balancing? URL: https://www.nginx.com/resources/ glossary/round-robin-load-balancing/ (дата обращения: 09.05.2017).
6. Cassandra drivers and libraries URL: https:// www.slideshare.net/doanduyhai/cassandra-drivers-and-libraries (дата обращения: 18.05.2017).
7. DC Aware Failover Round Robin Policy for Datastax Java Driver. URL: https://issues.sonatype.org/browse/OSSRH-17483 (дата обращения: 18.05.2017).
8. Token-aware Load Balancing Policy. URL: http://datastax.github.io/ruby-driver/features/load_balancing/token_aware/ (дата обращения: 17.05.2017).
9. Token Load Balancing Method. URL: https://www.citrix.com/blogs/2011/06/10/token-load-balancing-method/ (дата обращения: 18.05.2017).
10. Use Case 2: Configuring the Token Method of Load Balancing for DataStream. URL: https://docs.citrix.com/en-us/netscaler/11/traffic-management/datastream/configure-token-method-for-datastream.html (дата обращения: 19.05.2017).
11. Managed File Transfer and Network Solutions. URL: http://www.jscape.com/blog/load-balancing-algorithms. (дата обращения: 19.05.2017).
12. Обзор распределения нагрузки в LVS. URL: http://www.rhd.ru/docs/manuals/enterprise/RHEL-AS-2.1-Manual/install-guide/s1-lvs-scheduling.html (дата обращения: 19.05.2017).

13. The Least Response Time Method. URL: https://docs.citrix.com/en-us/netscaler/11/traffic-management/load-balancing/load-balancing-customizing-algorithms/leastresponsetime-method.html (дата обращения: 20.05.2017).
14. K6406: Overview of Least Connections, Fastest, Observed, and Predictive load balancing modes. URL: https://support.f5.com/csp/article/K6406 (дата обращения: 19.05.2017).
15. Tech Tip: Perceptive Load Balancing. URL: http://community.brocade.com/t5/vADC-Docs/Tech-Tip-Perceptive-Load-Balancing/tap/73724 (дата обращения: 19.05.2017).
16. DNS Support for Load Balancing. URL: https://tools.ietf.org/html/rfc1794 (дата обращения: 18.05.2017).

DOI: 10.17223/9785946216531/14

Использование алгоритмов обработки данных мобильных видеоизмерений для распознавания и классификации повреждений дорожной поверхности*

С.А. Проханов[1], Б.М. Шумилов[2], А.Ж. Кудуев[3], К.А. Абдыкалыков[3]

[1] Томский государственный университет,
[2] Томский государственный архитектурно-строительный университет,
[3] Ошский государственный университет
[1]viking@math.tsu.ru, [2] sbm05@yandex.ru, [3] altynbek_kuduev@mail.ru,
[4] kun.abd@mail.ru

Решение задачи разбивается на несколько этапов: 1) фильтрация и цветоделение последовательных кадров видеосъемки; 2) поиск локальных особенностей изображения на отдельных видеокадрах; 3) выявление устойчивых особенностей и формирование по ним опорных кадров; 4) обратное преобразование стереометрии; 5) создание пространственной триангуляционной сети; 6) получение трехмерной математической модели объекта. В данной работе мы анализируем известный алгоритм восстановления трехмерных координат точек объекта по двум смежным изображениям стереопары.
Ключевые слова: фильтрация изображений, мобильные видеоизмерения, фотограмметрия, 3D моделирование, автомобильные дороги.

Мобильная видеосъемка традиционно используется при «ручной» диагностике и паспортизации текущего состояния автомобильных дорог. В последние годы вычислительные возможности мобильных устройств существенно выросли, что позволяет поставить задачу диагностики повреждений дорожной поверхности по измерениям на серии последовательных кадров видеосъемки. В плоском случае решение обратной задачи фотограмметрии с помощью расстановки графических маркеров не встречает особенных трудностей (см. [1]). В пространственном случае [2] решение задачи сводится к формированию по графическим меткам элементов матрицы геометрических преобразований и восстановлению трехмерных координат точек объекта по перспективным проекциям в каждой картинной плоскости.

* Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований и Администрации Томской области (код проекта 16-41-700400 р_а).

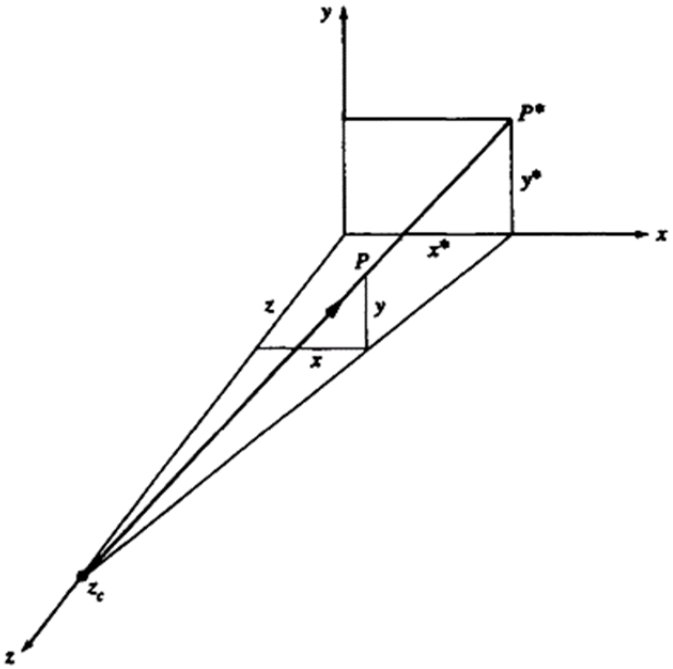


Рис. 1. Перспективная проекция точки

Напомним, что с введением однородных геометрических координат преобразование линейной перспективы (рис. 1) представляется в виде матрицы размера 4x4: $[x \quad y \quad z \quad 1][T'] = [x' \quad y' \quad z' \quad h]$, где

$$[T'] = \begin{bmatrix} T'_{11} & T'_{12} & T'_{13} & T'_{14} \\ T'_{21} & T'_{22} & T'_{23} & T'_{24} \\ T'_{31} & T'_{32} & T'_{33} & T'_{34} \\ T'_{41} & T'_{42} & T'_{43} & T'_{44} \end{bmatrix}.$$

В процессе фотографирования результаты проецируются на двумерную плоскость, в данном случае, на плоскость $z = 0$ с помощью преобразования проекции

$$[T''] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Последовательное применение линейных преобразований векторного пространства можно рассматривать как произведение матриц. Таким образом, результирующее преобразование можно записать в виде

$$\begin{bmatrix} x & y & z & 1 \end{bmatrix}[T] = \begin{bmatrix} x' & y' & 0 & h \end{bmatrix} = h\begin{bmatrix} x^* & y^* & 0 & 1 \end{bmatrix}. \qquad (1)$$

Здесь

$$[T] = [T''][T'] = \begin{bmatrix} T'_{11} & T'_{12} & 0 & T'_{14} \\ T'_{21} & T'_{22} & 0 & T'_{24} \\ T'_{31} & T'_{32} & 0 & T'_{34} \\ T'_{41} & T'_{42} & 0 & T'_{44} \end{bmatrix},$$

x^* и y^* – это координаты перспективной проекции на картинную плоскость фотоизображения $z = 0$. После исключения масштабного коэффициента h получим два скалярных уравнения:

$$\left(T'_{11} - T'_{14}x^*\right)x + \left(T'_{21} - T'_{24}x^*\right)y + \left(T'_{31} - T'_{34}x^*\right)z + \left(T'_{41} - T'_{44}x^*\right) = 0, \qquad (2a)$$

$$\left(T'_{11} - T'_{14}x^*\right)x + \left(T'_{21} - T'_{24}y^*\right)y + \left(T'_{31} - T'_{34}y^*\right)z + \left(T'_{41} - T'_{44}y^*\right) = 0, \qquad (2b)$$

Наконец, в последнем подходе [3,4] полагаются известными $[T], x^*, y^*$. В этом случае получается два уравнения от трех неизвестных пространственных координат x, y, z. Это – недоопределенная система уравнений, поэтому решить ее невозможно. Однако, если известны две перспективные проекции, скажем две фотографии, полученные с разных ракурсов, то уравнения (2) можно записать для обеих проекций. В результате получим

$$\left(T'^1_{11} - T'^1_{14}x^{*1}\right)x + \left(T'^1_{21} - T'^1_{24}x^{*1}\right)y + \left(T'^1_{31} - T'^1_{34}x^{*1}\right)z = x^{*1} - T'^1_{41},$$

$$\left(T'^1_{12} - T'^1_{14}y^{*1}\right)x + \left(T'^1_{22} - T'^1_{24}y^{*1}\right)y + \left(T'^1_{32} - T'^1_{34}y^{*1}\right)z = y^{*1} - T'^1_{42},$$

$$\left(T'^2_{11} - T'^2_{14}x^{*2}\right)x + \left(T'^2_{21} - T'^2_{24}x^{*2}\right)y + \left(T'^2_{31} - T'^2_{34}x^{*2}\right)z = x^{*2} - T'^2_{41},$$

$$\left(T'^2_{12} - T'^2_{14}y^{*2}\right)x + \left(T'^2_{22} - T'^2_{24}y^{*2}\right)y + \left(T'^2_{32} - T'^2_{34}y^{*2}\right)z = y^{*2} - T'^1_{42},$$

где верхние индексы 1 и 2 обозначают первую и вторую перспективные проекции. Эти уравнения представляют четыре уравнения от трех неизвестных пространственных координат x, y, z. Таким образом, снова получена переопределенная система уравнений, и для нахождения решения можно применить методы наименьших квадратов и квадратного корня.

С целью иллюстрации метода рассмотрим реальную стереопару (рис. 2).



Рис. 2. Стереопара из двух фотографий дорожного конуса на фоне повреждения дорожной поверхности

Решения отыскивались в системе MathCAD. Координаты вершин 3D-объекта измерялись линейкой. Координаты соответствующих точек на изображениях фиксировались в графическом редакторе с использованием манипулятора «мышь». Попытка восстановления 3D-координат вершин объекта дала неплохое совпадение с исходными значениями. Более того, в ходе расчетов удалось поправить измерения на фото и опечатки при снятии отсчетов. Задаваясь измеренными значениями фиксированной точки поверхности дорожного полотна на левом и правом снимках, можно проверить выполнение теоремы Пифагора. В нашем случае вычисленное расстояние от вершины конуса до асфальтового покрытия составило 31,975 см, что на 0,078% отличается от паспортного значения 32 см.

В докладе предлагается следующий алгоритм создания интерфейса для реализации полученных алгоритмов на смартфоне под управлением ОС Андроид.

• Последовательно левый, а затем правый снимок стереопары сбрасываются на экран смартфона (рис. 3).

• На снимок накладывается изображение проволочной модели 3D объекта.

• На модели последовательно вспыхивают характерные точки объекта, и подается команда пользователю (дублируемая голосом) движением графического курсора выбрать соответствующую точку на фотографии.

• После обработки обоих снимков стереопары в памяти смартфона формируются матрицы преобразований линейной перспективы (операция так называемой калибровки).

• Далее подается голосовая команда пользователю выбрать произвольную точку на каждом снимке стереопары.

• Предлагается выбрать из соответствующего меню режим тестирования, (если это – характерная точка, принадлежащая объекту), либо режим проверки выполнения теоремы Пифагора (если это – точка на дорожном основании), либо переход к продолжению работы.

• Далее подается голосовая команда пользователю выбрать вторую точку на каждом снимке стереопары.

• Вычисляется расстояние между выбранными точками (еще одно тестирование).

• Далее пользователю предлагается выбрать третью и так далее точки на каждом снимке стереопары.

• По мере пополнения массива точек программа строит триангуляцию интересующей пользователя геометрической фигуры, для которой затем вычисляются площадь и объем данного конкретного повреждения дорожной поверхности.

Дальнейшая задача состоит в подготовке заказа-наряда на выполнение ремонтных работ по ликвидации данного конкретного дорожного дефекта.

В развитие данной темы в докладе ставится задача автоматизации процесса поиска графических маркеров и опорных точек на фотоизображениях. В качестве предварительного вывода отметим, что на изображениях объектов искусственного происхождения, например, дорожного конуса, в вершинах которого располагаются графические маркеры, характерно наличие углов, образованных пересечениями направляющих линий (кривых либо прямых). Поэтому для анализа таких изображений целесообразно применение так называемых «угловых» фильтров [5–7].

Объектам естественного происхождения таким, как промоины, ямы и выбоины в асфальтово-бетонном покрытии, присуще наличие локальных неоднородностей, поэтому для поиска локальных особенностей изображений с точки зрения последующей корректной идентификации точек

интерполяции трехмерного объекта в большей степени пригодны детекторы блобов (blob – с англ. капля), основанные на методе Лапласа [8].



Рис. 3. Имитация работы с изображением на экране смартфона

При этом наличие прямоугольной таблицы изображений позволяет эффективно применить к решению поставленных задач современную теорию шеарлетов [9–16].

Литература

1. Marker Meter – измерение размеров по фотографии в лучших традициях криминалистики. URL: http://4pda.ru/2015/06/04/224700 (дата обращения: 03.11.2016).
2. Роджерс Д., Адамс Дж. Математические основы машинной графики. М. : Мир, 2001. 604 с.
3. Sutherland I. E. Three Dimensional Data Input by Tablet // Proc. IEEE. 1974. Vol. 62, No. 2. Pp. 453–461.

4. Slama C. (ed.). Manual of Photogrammetry, American Society of Photogrammetry, 1980.
5. Harris C., Stephens M. A combined corner and edge detector // Fourth Alvey Vision Conference. Manchester, UK, 1988. Pp. 147–151.
6. Sonka M., Hlavac V., Boyle R. Image Processing, Analysis and Machine Vision. Thomson, 2008. 866 p.
7. Harris affine region detector. URL: https://en.wikipedia.org/wiki/Harris_affine_region_detector (дата обращения: 03.11.2016).
8. Tinne Tuytelaars, Krystian Mikolajczyk. Local Invariant Feature Detectors: A Survey, 2008.
9. Dahlke S. et al. Shearlet coorbit spaces and associated Banach frames // Applied and Computational Harmonic Analysis. 2009. V. 27, No. 2. P. 195–214.
10. Kittipoom P., Kutyniok G., Lim W. Q. Irregular shearlet frames: Geometry and approximation properties // Journal of Fourier Analysis and Applications. 2011. V. 17, No. 4. P. 604–639.
11. Kutyniok G. Construction of regular and irregular shearlets // Affine Density in Wavelet Analysis. 2007. P. 1–10.
12. Houska R. The nonexistence of shearlet scaling functions // Applied and computational harmonic analysis. 2012. V. 32, No. 1. P. 28–44.
13. Easley G., Labate D., Lim W.Q. Sparse directional image representations using the discrete shearlet transform // Applied and Computational Harmonic Analysis. 2008. V. 25, No. 1. P. 25–46.
14. Kutyniok G., Sauer T. Adaptive directional subdivision schemes and shearlet multiresolution analysis // SIAM Journal on Mathematical Analysis. 2009. V. 41, No. 4. P. 1436–1471.
15. Han B., Kutyniok G., Shen Z. Adaptive multiresolution analysis structures and shearlet systems // SIAM Journal on Numerical Analysis, 2011. V. 49, No. 5. P. 1921–1946.
16. Kutyniok G., Shahram M., Donoho D. L. Development of a digital shearlet transform based on pseudo-polar FFT // SPIE Optical Engineering Applications. International Society for Optics and Photonics. 2009. P. 74460B-74460B-13.

DOI: 10.17223/9785946216531/15

Разработка программных средств для обработки больших временных рядов

В.А. Тартаковский[1], И.А. Ботыгин[1,2], А.И. Шерстнёва[2]

[1] Институт мониторинга климатических и экологических систем СО РАН, Томск
[2] Томский политехнический университет, Томск
trtk@list.ru, bia@tpu.ru, sherstneva@tpu.ru

Описывается разработка методов, алгоритмов и программных средств вычислительной среды для параллельной обработки многомерных данных и больших временных рядов. На основе описания сетевых месторасположений климатических и иных параметров данные берутся непосредственно в режиме реального времени с серверов мировых центров научных данных. При этом используются реальные представления форматов хранения параметров для создания интерфейсных средств получения выборки научных данных, определенной исследователем. На компьютере исследователя сохраняется только результат указанной выборки. Дополнительно разработанная вычислительная среда позволяет получить первичную статистическую обработку выборки из временных рядов наблюдений, что облегчает дальнейший анализ данных.
Ключевые слова: большие данные, визуализация данных, мировые центры данных, картограммы, формат GRIB, формат NetCDF.

В настоящее время существует большое количество различных центров и ассоциаций, которые осуществляют мониторинг различных физических явлений и процессов, происходящих по всему земному шару. Важно, что при изучении и специальной обработке полученных данных появляется возможность прогнозировать различные явления. В ходе исследования были рассмотрены следующие крупнейшие мировые центры хранения научных данных: Национальное управление океанических и атмосферных исследований министерства торговли США [1], объединение университетов в области исследований атмосферы [2], Центр анализа экологических данных [3], Институт океанографии Скриппса [4], Европейский архив климатических данных (ECA&D) [5], Глобальные морские сети [6, 7].

Анализ организации и содержания климатических данных в указанных центрах показал определенное различие между структурами хране-

ния данных и особенностями форматов данных. Вместе с тем, наименования климатических параметров, станций и временные интервалы наблюдений могут совпадать. Это дает возможность разработать обобщенную схему доступа и получения, требуемых исследователю природно-климатических данных.

Обобщенная схема доступа и получения природно-климатических данных в режиме онлайн базируется на концепции единого логического подхода к их обработке. Для работы с данными необходимо выполнить ряд общих действий:

– проанализировать способы получения данных (выбор источников, центров, сервисов);

– проанализировать типы данных (определение нужного типа данных, исследование выбранного типа данных, анализ содержимого, определение переменных и констант);

– провести собственно обработку выбранных по области, дате, времени и типу природно-климатических данных.

Таким образом, для динамического автоматизированного получения данных из Интернет-ресурсов метеорологических центров и ассоциаций предлагается следующий алгоритм получения данных, основанный на обобщенной схеме доступа и получения природно-климатических данных:

Шаг 1. Система выводит список доступных метеорологических центров.

Шаг 2. Исследователь производит выбор метеорологического центра.

Шаг 3. Система выводит список доступных станций и областей, имеющихся в центре.

Шаг 4. Исследователь производит выбор станции, с которой велась фиксация данных или определенной области.

Шаг 5. Система выводит список доступных климатических параметров.

Шаг 6. Исследователь производит выбор интересующего климатического параметра или параметров.

Шаг 7. Система отображает доступные даты или временные промежутки для имеющихся климатических параметров.

Шаг 8. Исследователь производит выбор доступного временного интервала.

Шаг 9. Система обращается к ресурсу и получает выбранные данные.

Шаг 10. После получения данных в формате хранения производится автоматическая конвертация данных в понятный пользователю вид.

Функциональная структура разработанного программного комплекса представлена на рис. 1.

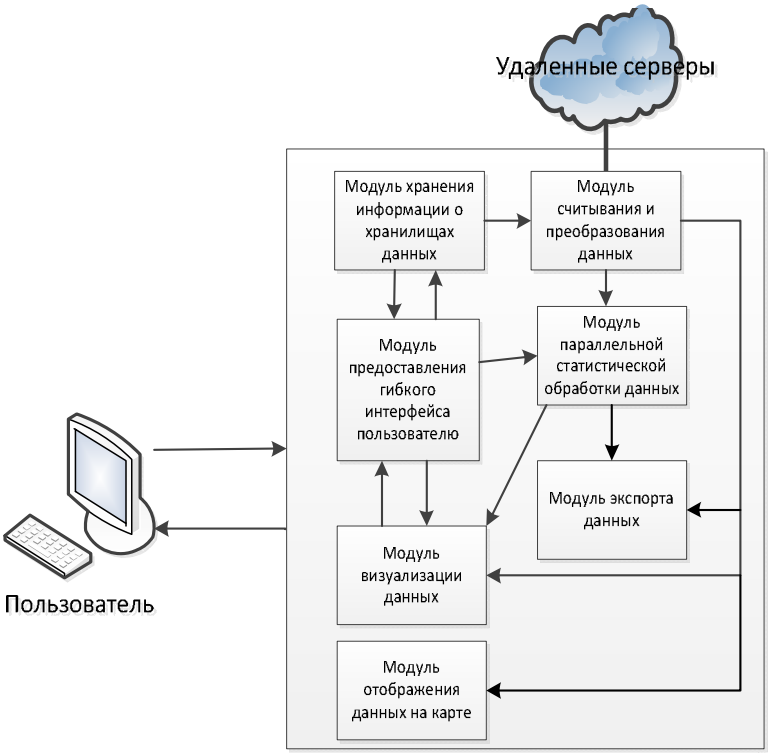


Рис. 1. Укрупненная функциональная структура программного комплекса для обработки больших временных рядов

Практическая апробация разработанного программного комплекса осуществлялась при работе с широко распространенными форматами хранения научных данных – GRIB, NetCDF и форматом хранения данных, используемым в ультразвуковом термоанемометре.

Ниже приведены некоторые практические результаты при работе с данными по морским волнам. Рассмотрены период волны и средняя высота волны в Карибском регионе за временной интервал с 01.06.2017 г. по 08.06.2017 г.

Интерфейс доступа представлен на рис. 2.

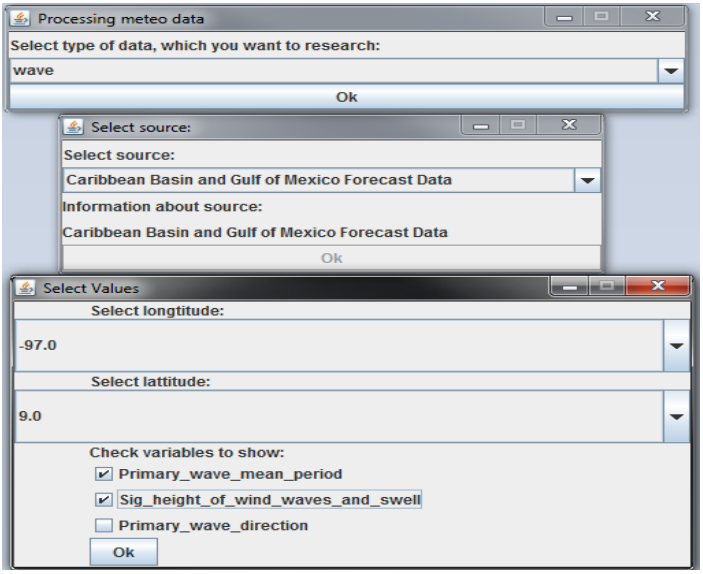


Рис. 2. Интерфейс выборки данных

Визуализация выбранных данных в точке (–77.0, 29.0) представлена на рис. 3.

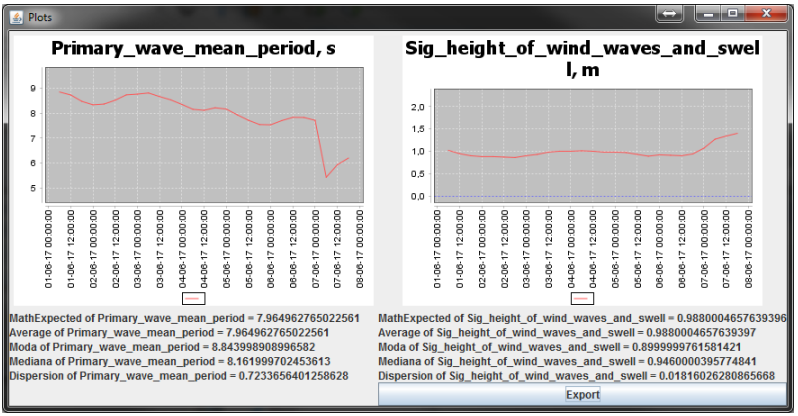


Рис. 3. Визуализация выбранных данных

В ходе исследования были решены следующие задачи: а) разработано формализованное описание сетевых месторасположений климатических и иных данных с указанием формата хранения данных; б) использование реального представления данных для создания алгоритмов и интерфейсных средств получения выборки, определенной исследователем; в) разработаны алгоритмы визуального анализа и экспорта полученных данных; г) реализована первичная статистическая обработка выборок из временных рядов наблюдений.

Таким образом, разработанные алгоритмы и методы обработки данных позволяют исследователю работать с большими файлами временных рядов независимо от хранимых параметров. Данные берутся непосредственно в режиме реального времени с серверов мировых центров данных, а на компьютере исследователя сохраняется только результат указанной выборки.

Литература

1. Free World-Wide climate, weather, etc. data. URL: http://www.noaa.gov/ (дата обращения: 22.05.2017).
2. CEDA website. URL: http://badc.nerc.ac.uk/ (дата обращения: 22.05.2017).
3. SCRIBBS Institution of Oceanography. URL: https://scripps.ucsd.edu/ (дата обращения: 12.05.2017).
4. University Corporation for Atmospheric Research. URL: https://www2.ucar.edu/ (дата обращения: 22.05.2017).
5. Global Marine Networks Website. URL: http://www.globalmarinenet.com (дата обращения: 22.05.2017).
6. A guide to the code form FM 92-IX Ext. GRIB. URL: https://www.wmo.int/pages/prog/www/WDM/Guides/Guide-binary-2.html (дата обращения: 22.05.2017).
7. Global Surface Summary of the Day – GSOD. URL: https://data.noaa.gov/dataset/global-surface-summary-of-the-day-gsod (дата обращения: 22.05.2017).

Научное издание

ДЕВЯТАЯ СИБИРСКАЯ КОНФЕРЕНЦИЯ
по параллельным и высокопроизводительным вычислениям

Томск, 10−12 октября 2017 года

Сборник статей

Издание подготовлено в авторской редакции
Оригинал-макет А.И. Лелоюр

Подписано к печати 29.11.2016 г. Формат $60\times84^{1}/_{16}$.
Бумага для офисной техники. Гарнитура Times.
Усл. печ. л. 6,4.
Тираж 250 экз. Заказ № 2865.

Отпечатано на оборудовании
Издательского Дома
Томского государственного университета
634050, г. Томск, пр. Ленина, 36
Тел. 8+(382-2)–52-98-49
Сайт: http://publish.tsu.ru
E-mail: rio.tsu@mail.ru

ISBN 978-5-94621-653-1

