

PARALLEL NUMERICAL METHOD FOR SOLUTION OF HYDRODYNAMIC EQUATIONS IN THE SHALLOW WATER APPROXIMATION FOR SHARED MEMORY COMPUTERS

A. V. Starchenko

Tomsk State University,
634050, Tomsk, Russia
Federal Research Center for informational and computational technologies,
630090, Novosibirsk, Russia

DOI: 10.24412/2073-0667-2024-1-41-56

EDN: JVZEVV

Modeling of natural phenomena such as tsunamis, floods, ocean and river currents, and dam breaks are pressing environmental problems. Due to the significant difference in the vertical and horizontal dimensions of the study area, non-stationary two-dimensional hydrodynamic equations in the shallow water approximation have become very popular among researchers of the processes under consideration using mathematical modeling methods, which makes it possible to significantly simplify the mathematical formulation of the problems under consideration. Note, that the accuracy of numerical prediction using such equations depends on a number of conditions, among which the main ones are the spatial resolution of the study area and the quality of the selected numerical methods. However, increasing the spatial resolution significantly increases the time required to obtain a numerical solution, since in the numerical modeling of processes in the environment, as a rule, explicit or semi-implicit difference schemes are used with a limitation on the time step, depending on the size of the spatial grid steps. That is, as a result, due to the need to carry out calculations with a large number of grid nodes and a smaller time step, calculations on a computer with a sequential architecture can take quite a long time.

The aim of this work is to formulate a mathematical problem of unsteady isothermal turbulent flows in river, based on the shallow water approximation, to construct an effective conservative numerical method, and to develop parallel computing algorithms for multi-core computing systems with shared RAM.

The turbulent isothermal motion of an incompressible Newtonian fluid in a river flow is considered. The study area is an open river channel with islands and irregular bottom topography. Current characteristics can vary significantly over time, and therefore the current is considered unsteady with potential flooding of coastal areas. The movement of water in a river is determined by the forces of gravity and friction. In addition, the influence of the Coriolis force and turbulent diffusion on the flow is taken in to account. It is assumed that the horizontal dimensions of the region significantly exceed the vertical ones, the Reynolds-averaged flow characteristics change slightly in the vertical direction, and the vertical pressure distribution is hydrostatic. The thermophysical properties of water (viscosity, density) are considered constant. For numerical modeling of unsteady isothermal turbulent flows in river flows, a mathematical model is formulated based on the shallow water approximation for the Reynolds equations for an incompressible fluid. To take into account the transport, generation, diffusion and dissipation of turbulent vortices, this work uses the $k - \varepsilon$ turbulence model constructed by Rastogi

and Rodi from the original $k - \varepsilon$ model of Launder and Spalding to close the Reynolds shallow water equations.

To construct a discrete analogue of the developed mathematical model, the rectangular computational domain in which the flow with variable boundaries is studied is covered with a structured mesh with steps $\Delta x, \Delta y$, respectively. According to the concept of the finite volume method, each internal mesh node appears in a separate finite volume. In this case, the values of the flow depth, bottom topography (and turbulence model parameters) are determined at the nodes of the computational grid, and the velocity vector components are determined at the midpoints of the corresponding faces of the finite volumes. The differential equations of the model are integrated over each finite volume. The equations of motion are approximated in cells shifted by $\Delta x/2, \Delta y/2$, respectively. For approximate calculation of integrals, quadrature formulas of average rectangles are used, derivatives are approximately calculated using central-difference formulas. To calculate fluxes on the faces of finite volumes, monotonized upstream van Leer approximations [1] with a limiter [2] are used. When approximating equations in time, conditionally stable semi-implicit difference schemes are used, which are conservative not only for the continuity equation, but also for the equations of motion, which is very important for obtaining non-negative values of flow depth [3]. As a result, semi-implicit difference schemes of first order approximation in time and second order in spatial coordinates are obtained.

The classical two-dimensional Thacker problem of fluid oscillations without friction in a reservoir whose bottom is a paraboloid was considered as a problem on which the accuracy of calculations and the quality of parallelization were tested. Satisfactory agreement between the calculated values of the flow depth and the analytical solution was obtained.

The constructed semi-implicit difference scheme on staggered difference grids has great potential for creating an efficient parallel program, since at each time step, calculations of depth or velocity components can be performed simultaneously and independently in each computational node of the grid. On a server with a total memory of 192 GB and two twelve-core Intel Xeon Silver 4214 2.2 GHz processors and one NVIDIA GeForce RTX2080 Ti graphics card, a study was conducted of the effectiveness of parallel programs built using low-cost parallel programming technologies Open Multi-Processing and Open Accelerators. In these programs, using OpenMP or OpenACC parallel technologies, the execution of iterations of nested loops was evenly distributed among the active threads/processes of the central or graphic processor cores.

The calculation using a sequential program for the conditions under consideration on the server was 547.7 s (500×500 grid) and 4606.4 s (1000×1000 grid). Parallel calculations showed that the use of OpenMP technology for two twelve-core central processors makes it possible to speed up the computing process by more than 15 times. Using OpenACC technology when calculating on the same multi-core system and the NVIDIA GeForce RTX2080 Ti graphics processor gives a speedup of more than 25.

Key words: parallel computations, shallow water equations, shared memory computers, OpenMP, OpenACC.

References

1. Saint-Venant A. J. B. Theorie du Mouvement non permanent des Eaux // Institut de France, Acad. des Sci. de Paris. 1871. 73 (3) 147; 73 (4) 237.
2. Gusev O. I., Khakimzyanov G. S., Skiba V. S., Chubarov L. B. Shallow water modeling of wave-structure interaction over irregular bottom // Ocean Engineering. 2023. V. 267. Art. 113284.
3. Brand S. Parallel algorithm for numerical solution of the shallow water equation // Proceedings of the Czech-Japanese Seminar in Applied Mathematics. 2006. Czech Technical University in Prague, September 14–17, 2006. P. 25–36.
4. Gabdullina M. I. Parallel algorithm for numerical solution of the two-layer shallow water model in 2D // Russian Supercomputing Days 2016 // RussianSCDays.org. P. 940–947.

5. Chaplygin A.V., Gusev A.V. Shallow water model using a hybrid MPI/OpenMP parallel programming // Problems of informatics, 2021. N 1. P. 65–82.
6. Juliati S., Gunawan P.H. OpenMP architecture to simulate 2D water oscillation on paraboloid // 5th International Conference on Information and Communication Technology (ICoIC7), 2017. P. 1–5.
7. Liu Q., Qin Y., Li G. Fast Simulation of Large-Scale Floods Based on GPU Parallel Computing // Water. 2018. V. 10(5):589.
8. Zhang S., Li W., Jing Zh, Yi Y., Zhao Y. Comparison of Three Different Parallel Computation Methods for a Two-Dimensional Dam-Break Model // Mathematical Problems in Engineering. 2017. V. 2017, Article ID 1970628, 12 pages.
9. Arnoldy A., Adytia D. Performance of Staggered Grid Implementation of 2D Shallow Water Equations using CUDA Architecture // 2019 12th International Conference on Information & Communication Technology and System (ICTS), Surabaya, Indonesia, 2019, P. 286–290.
10. Churuksaeva V., Starchenko A. Mathematical modeling of a river stream based on a shallow water approach // Procedia Computer Science. 2015. V. 66. P. 200–209.
11. Rastogi A.K., Rodi W. Predictions of heat and mass transfer on open channels // ASCE Journal of the Hydraulics Division. 1978. V. 104(3). P. 397–420.
12. Launder B.E., Spalding D.B. The numerical computation of turbulent flows // Computer Methods in Applied Mechanics and Engineering. 1974. V. 2(3). P. 269–289.
13. Stelling G.S., Duimijer S.P.A. A staggered conservative scheme for every Froude number in rapidly varied shallow water flows // Int. Journal for Numerical Methods in Fluids. 2003. V. 43. P. 1329–1354.
14. van Leer B. Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method // Journal of Computational Physics. 1979. V. 32(1). P. 101–136.
15. Cada M., Torrilhon M. Compact third-order limiter functions for finite volume methods // Journal of Computational Physics. 2009. V. 228. P. 4118–4145.
16. Thacker W.C. Some exact solutions to the nonlinear shallow water wave equations // Journal of Fluid Mechanics. 1981. V. 107. P. 499–508.

ПАРАЛЛЕЛЬНЫЙ ЧИСЛЕННЫЙ МЕТОД РЕШЕНИЯ ГИДРОДИНАМИЧЕСКИХ УРАВНЕНИЙ В ПРИБЛИЖЕНИИ МЕЛКОЙ ВОДЫ ДЛЯ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ С ОБЩЕЙ ПАМЯТЬЮ

А. В. Старченко

Томский государственный университет,
634050, Томск, Россия
ФИЦ информационных и вычислительных технологий,
630090, Новосибирск, Россия

УДК 519.63, 519.683, 519.688

DOI: 10.24412/2073-0667-2024-1-41-56

EDN: JVZEVV

Для численного моделирования нестационарных изотермических турбулентных течений в речных потоках сформулирована математическая модель, опирающаяся на приближение мелкой воды для уравнений Рейнольдса для несжимаемой жидкости, эффективный численный метод, обеспечивающий в рамках использования метода конечного объема, структурированных разнесенных сеток и полунеявных разностных схем выполнение на разностном уровне законов сохранения массы и импульса. Вычислительная реализация предложенной модели и метода была протестирована на аналитическом решении Такера и распараллелена с помощью технологий OpenMP и OpenACC на гибридной многоядерной системе с общей памятью. Расчеты показали, что использование технологии OpenMP для двух двенадцатиядерных центральных процессоров позволяет более чем в 15 раз ускорить вычислительный процесс. Использование технологии OpenACC при расчетах на этой же многоядерной системе и графическом процессоре NVIDIA GeForce RTX2080Ti дает ускорение более чем в 25.

Ключевые слова: параллельные вычисления, уравнения мелкой воды, системы с общей памятью, OpenMP, OpenACC.

Введение. Моделирование таких природных явлений как цунами, наводнения, океанические и речные течения, прорыв дамбы являются актуальными проблемами окружающей среды. В силу существенного различия в вертикальных и горизонтальных размерах области большую популярность у исследователей рассматриваемых процессов с помощью методов математического моделирования приобрели нестационарные двумерные гидродинамические уравнения в приближении мелкой воды [1], позволяющие значительно упростить математическую постановку проблем. Заметим, однако, что точность численного прогноза с помощью таких уравнений зависит от ряда условий, среди которых основными являются пространственное разрешение области исследования и качество выбранных численных методов. Однако повышение пространственного разрешения существенно увеличивает временные затраты на получение численного решения, поскольку при численном моделировании процессов в окружающей среде, как правило, используются явные или полунеявные разностные схемы с ограничением на шаг по времени, зависящим от

размера шагов пространственной сетки. То есть в итоге из-за необходимости вести вычисления с большим количеством узлов сетки и меньшей величиной шага по времени расчеты на компьютере с последовательной архитектурой могут занимать достаточно долгое время [2–3]. Для ускорения вычислений с высоким пространственным разрешением целесообразно использовать параллельные компьютерные технологии, которые очень важны и крайне необходимы. В настоящее время на современных многоядерных многопроцессорных вычислительных системах наиболее популярными являются следующие технологии [3–9]: Message Passing Interface (MPI), Open Multi-Processing (OpenMP), OpenAccelerators (OpenACC), Compute Unified Device Architecture (CUDA). С учетом этого параллельная реализация численных моделей, основанных на уравнениях мелкой воды, может быть разделена на три категории:

- модели, ориентированные на массивные вычислительные системы с многоядерными узлами и распределенной памятью, используют, как правило, технологии параллельного программирования MPI или MPI+OpenMP [3, 5, 8];

- модели, ориентированные на многоядерные вычислительные системы с общей оперативной памятью, используют технологию OpenMP [3, 6, 8];

- модели, ориентированные на гибридные вычислительные системы с центральными и графическими многоядерными процессорами (CPU+GPU) и общей оперативной памятью, используют технологии CUDA или OpenACC [7–9].

При использовании вычислительных систем с общей оперативной памятью обмен данными между параллельно исполняемыми нитями программы осуществляется путем чтения и записи информации прямо в оперативной памяти, что позволяет легче решать проблемы балансировки общей вычислительной нагрузки между активными нитями. В технологии OpenMP основным средством балансировки нагрузки является распределение итераций вложенных циклов между активными нитями/процессами. Параллельные программы, развиваемые на основе этого подхода, имеют преимущества в простоте разработки кода, хорошей масштабируемости и переносимости. Например, в [3] уравнения мелкой воды решаются методами конечных разностей и конечных объемов в рамках схем Лакса-Фридрикса, Лакса-Вендроффа и Маккормака. Алгоритмы распараллелены с использованием OpenMP и MPI. Результаты измерения эффективности параллельных программ показывают, что OpenMP, так же как MPI подходят для распараллеливания программ численного решения уравнения мелкой воды, поскольку значения эффективности для сеток 400×400 , \dots , 2000×2000 не опускались ниже 85 % во время тестирования на относительно небольшом количестве ядер (4 и больше).

В MPI-параллельных программах численного решения уравнений мелкой воды распределение вычислений между отдельными многопроцессорными многоядерными вычислительными узлами с локальной оперативной памятью, связанными компьютерной сетью, целиком лежит на разработчике параллельного программного обеспечения. Это с одной стороны значительно повышает трудоемкость программирования, с другой — открывает дополнительные возможности масштабируемости параллельных приложений за счет возможности привлечения большего количества вычислительных ресурсов (CPUs), чем в случае вычислительных систем с общей памятью. Заметим, однако, что такие вычислительные системы кластерного типа встречаются гораздо реже, чем сервера с 1–2 CPU и общей оперативной памятью. В последнее время для моделирования океанических течений в приближении мелкой воды были получены успешные результаты в совместном привлечении двух технологий параллельного программирования: MPI+OpenMP. Напри-

мер, в [5] был разработан гибридный задачный MPI+OpenMP подход решения уравнений мелкой воды с помощью аппроксимаций второго порядка на разнесенных сетках и явной численной схемы «чехарда», обеспечивающий на сетке 1525×1115 эффективность более 90 % (несколько выше, чем чистый MPI) при использовании 144 вычислительных ядер.

Среди всех параллельных программ, ориентированных на вычислительные системы с общей памятью, программы, использующие графические процессоры, дают наибольшее ускорение в параллельных вычислениях [7–9]. Наличие большего числа (до нескольких тысяч) производительных для расчетов с плавающей точкой графических процессоров открывает большие возможности для сокращения времени проведения большого объема вычислений [7–9]. Трудоемкость программирования в рамках технологии OpenACC незначительно отличается от трудоемкости использования технологии OpenMP, дополнительной необходимостью является обеспечение передачи информации между общей памятью компьютера и памятью графической карты. Несколько выше трудоемкость программирования CUDA-приложений, поскольку в ней подобно MPI, распределение вычислений в большей степени лежит на разработчике параллельной программы. Тем не менее, в [7] использование в вычислениях графической карты NVIDIA Tesla K20 и технологии OpenACC при моделировании реальных наводнений в Китае на треугольной сетке с количеством узлов 2868736 удалось достичь ускорения в 31 раз по сравнению с последовательными вычислениями на CPU Intel Xeon E5-2690 @ 3.0 GHz.

Целью данной работы является формулировка математической постановки задачи о нестационарных изотермических турбулентных течениях в речных потоках, опирающейся на приближение мелкой воды, построение эффективного консервативного численного метода, разработка параллельных вычислительных алгоритмов для многоядерных вычислительных систем с общей оперативной памятью.

1. Постановка задачи. Рассматривается турбулентное изотермическое движение несжимаемой ньютоновской жидкости в речном потоке. Область исследования представляет собой участок открытого речного русла с островами и нерегулярным рельефом дна. Характеристики течения могут значительно меняться со временем, и потому течение считается нестационарным. Движение воды в реке определяется силами гравитации и трения. Кроме того, учитывается влияние на течение силы Кориолиса и турбулентной диффузии.

Предполагается, что горизонтальные размеры области существенно превосходят вертикальные, осредненные по Рейнольдсу характеристики течения слабо меняются в вертикальном направлении, а вертикальное распределение давления является гидростатическим. Теплофизические свойства воды (вязкость, плотность) считаются постоянными.

1.1. *Система уравнений и краевые условия.* Математическая модель в приближении мелкой воды [10] включает уравнение неразрывности:

$$\frac{\partial h}{\partial t} + \frac{\partial(hu)}{\partial x} + \frac{\partial(hv)}{\partial y} = 0, \quad (1)$$

уравнения движения:

$$\begin{aligned} \frac{\partial(hu)}{\partial t} + \frac{\partial(hu^2)}{\partial x} + \frac{\partial(huv)}{\partial y} &= -gh \frac{\partial(z_b + h)}{\partial x} + \frac{1}{\rho} \frac{\partial(h\tau_{xx})}{\partial x} + \frac{1}{\rho} \frac{\partial(h\tau_{xy})}{\partial y} + \\ &+ \frac{(\tau_{xz})_s - (\tau_{xz})_b}{\rho} - hF_x, \\ \frac{\partial(hv)}{\partial t} + \frac{\partial(huv)}{\partial x} + \frac{\partial(hv^2)}{\partial y} &= -gh \frac{\partial(z_b + h)}{\partial y} + \frac{1}{\rho} \frac{\partial(h\tau_{yx})}{\partial x} + \frac{1}{\rho} \frac{\partial(h\tau_{yy})}{\partial y} + \\ &+ \frac{(\tau_{yz})_s - (\tau_{yz})_b}{\rho} - hF_y. \end{aligned} \quad (2)$$

Здесь $h(t,x,y)$ — глубина; $u(t,x,y)$, $v(t,x,y)$ — осредненные по глубине значения компонент вектора скорости; $z_b(x,y)$ — рельеф дна; ρ — плотность воды, g — ускорение свободного падения; τ_{xx} , τ_{xy} , τ_{yx} , τ_{yy} — осредненные по глубине компоненты тензора вязких напряжений и напряжений Рейнольдса; $(\tau_{xz})_s$, $(\tau_{xz})_b$, $(\tau_{yz})_s$, $(\tau_{yz})_b$ — трение на поверхности воды и на дне, соответственно. Предполагается, что в рассматриваемом случае ветровое трение не оказывает на речное течение значительного влияния и потому не учитывается в модели. Осредненные по глубине компоненты силы Кориолиса определяются следующим образом:

$$F_x = (-4\pi/\tau) \sin(lat)v, F_y = (4\pi/\tau) \sin(lat)u,$$

где lat — географическая широта, τ — продолжительность суток в секундах.

Для получения неизвестных значений компонент тензора напряжений в уравнениях модели используются градиентные соотношения Буассинеса:

$$\begin{aligned} \frac{\tau_{xy}}{\rho} = \frac{\tau_{yx}}{\rho} &= (\nu + \nu_t) \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right), \\ \frac{\tau_{xx}}{\rho} &= 2(\nu + \nu_t) \frac{\partial u}{\partial x} - \frac{2}{3}k, \\ \frac{\tau_{yy}}{\rho} &= 2(\nu + \nu_t) \frac{\partial v}{\partial y} - \frac{2}{3}k, \end{aligned} \quad (3)$$

где ν — молекулярная кинематическая вязкость воды, ν_t — турбулентная вязкость, k — кинетическая энергия турбулентности.

Трение о дно определяется как:

$$(\tau_{xz})_b = \rho c_f u |\vec{\mathbf{w}}|, (\tau_{yz})_b = \rho c_f v |\vec{\mathbf{w}}|,$$

где c_f — коэффициент трения, зависящий от физических характеристик русла или канала: $c_f = gn^2/h^{0,33}$ определяется из закона трения Маннинга, где n — коэффициент Маннинга, характеризующий шероховатость русла, $|\vec{\mathbf{w}}|$ — модуль вектора скорости.

Для учета переноса, генерации, диффузии и диссипации турбулентных вихрей в данной работе для замыкания уравнений мелкой воды применяется $k - \varepsilon$ модель турбулентности, построенная Растоги и Роди [11] из оригинальной $k - \varepsilon$ модели Лаундера и Сполдинга [12] для замыкания уравнений Рейнольдса.

На входе в расчетную область задаются однородные профили нормальной к границе компоненты скорости потока U_0 , энергии турбулентности k_0 , диссипации ε_0 и глубины

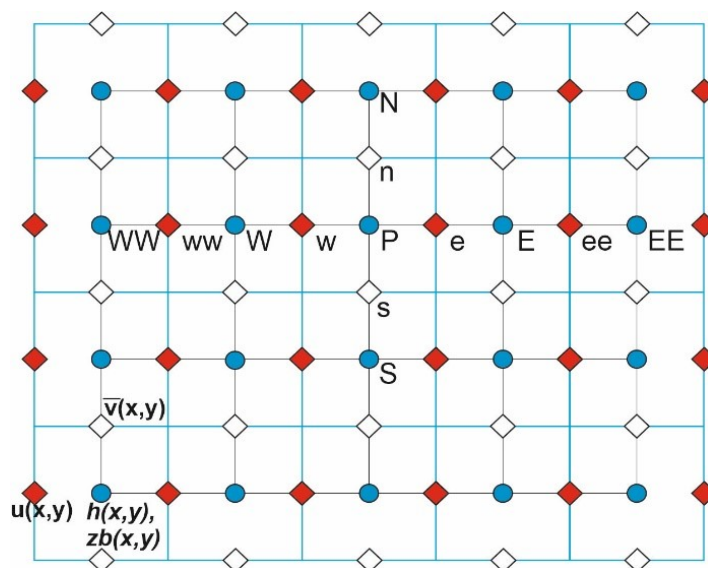


Рис. 1. Расчетная сетка. Круглые значки соответствуют центру конечного объема, заполненные ромбы — компоненте скорости u , открытые ромбы — v

потока h_0 , полученные из эмпирических данных. Поперечная компонента скорости потока на входе задается равной нулю. В качестве граничных условий на выходе из области используется равенство нулю производных по внешней нормали к выходной границе для основных характеристик течения. В начальный момент распределения глубины потока компонент скорости и параметров энергии турбулентности на рассматриваемой области считаются известными.

2. Численный метод. Для построения дискретного аналога разрабатываемой математической модели расчетная область размером $[0, L_x] \times [0, L_y]$, в которой исследуется течение потока с переменными границами, покрывается структурированной сеткой с шагами $\Delta x, \Delta y$, соответственно. Согласно концепции метода конечных объемов, каждый внутренний узел сетки оказывается в отдельном конечном объеме (рис. 1). При этом значения глубины потока, рельефа дна (и параметров модели турбулентности) определяются в узлах расчетной сетки, а компоненты вектора скорости — в серединах соответствующих граней конечных объемов. Для рассматриваемого фрагмента сетки используются следующие обозначения. Каждый конечный объем сетки имеет западную (w), восточную (e), северную (n) и южную (s) границы и центр (P). Центры соседних конечных объемов обозначаются W, E, N, S , соответственно.

Дифференциальные уравнения модели интегрируются по каждому конечному объему. Уравнения движения аппроксимируются в ячейках, сдвинутых на $\Delta x/2$ и $\Delta y/2$, соответственно. Для приближенного вычисления интегралов используются квадратурные формулы средних прямоугольников, производные приближенно вычисляются по центрально-разностным формулам. При аппроксимации уравнений по времени используются условно устойчивые полунеявные разностные схемы, консервативные не только для уравнения неразрывности (1), но и для уравнений движения (2), что весьма важно для получения неотрицательных значений глубины потока h [13].

В итоге получаются полунеявные разностные схемы первого порядка аппроксимации по времени и второго — по пространственным координатам.

$$\frac{h_p^{k+1} - h_p^k}{\Delta t_k} + \frac{G_e^k - G_w^k}{\Delta x} + \frac{G_n^k - G_s^k}{\Delta y} = 0, \quad (4)$$

$$\begin{aligned} & \frac{u_e^{k+1} - u_e^k}{\Delta t_k} + \frac{1}{h_e^k} \left[\frac{q_{xE}^k u_E^k - q_{xP}^k u_P^k}{\Delta x} - u_e^k \frac{q_{xE}^k - q_{xP}^k}{\Delta x} \right] + \\ & + \frac{1}{h_e^k} \left[\frac{q_{yne}^k u_{ne}^k - q_{yse}^k u_{se}^k}{\Delta y} - u_e^k \frac{q_{yne}^k - q_{yse}^k}{\Delta y} \right] + \\ & + g \frac{\xi_E^{k+1} - \xi_P^{k+1}}{\Delta x} + \frac{c_f |\vec{w}_e^k| u_e^{k+1}}{h_e^k} - (D_x)_e^k + (F_x)_e^k = 0, \end{aligned} \quad (5)$$

$$\begin{aligned} & \frac{v_n^{k+1} - v_n^k}{\Delta t_k} + \frac{1}{h_n^k} \left[\frac{q_{xEn}^k v_{en}^k - q_{xwn}^k v_{wn}^k}{\Delta x} - v_n^k \frac{q_{xEn}^k - q_{xwn}^k}{\Delta x} \right] + \\ & + \frac{1}{h_n^k} \left[\frac{q_{yN}^k v_N^k - q_{yP}^k v_P^k}{\Delta y} - v_n^k \frac{q_{yN}^k - q_{yP}^k}{\Delta y} \right] + \\ & + g \frac{\xi_N^{k+1} - \xi_P^{k+1}}{\Delta x} + \frac{c_f |\vec{w}_n^k| v_n^{k+1}}{h_n^k} - (D_y)_n^k + (F_y)_n^k = 0. \end{aligned} \quad (6)$$

Здесь $\xi = z_b + h$ — уровень поверхности воды, $(D_x)_e^k, (D_y)_n^k$ — аппроксимации диффузионных слагаемых уравнений (2); для вычисления потоков на гранях конечных объемов $G_e = (uh)_e = (q_x)_e, G_n = (vh)_n = (q_y)_n$ используются монотонизированные противопотоковые аппроксимации ван Лира [14] с ограничителем [15]:

$$G_e = \begin{cases} u_e(h_p + 0.5\Psi(\theta^+)(h_E - h_p)), & u_e > 0 \\ u_e(h_E + 0.5\Psi(\theta^-)(h_E - h_p)), & u_e \leq 0 \end{cases}, \quad \Psi(\theta) = \max\left(0, \min\left(\min\left(2\theta, \frac{\theta+2}{3}\right), 2\right)\right),$$

$$\begin{aligned} \theta_e^+ &= \frac{h_P - h_W}{h_E - h_P}, \\ \theta_e^- &= \frac{h_{EE} - h_E}{h_E - h_P} \end{aligned}$$

$$h_e = \frac{h_E + h_P}{2}, \quad q_{xP} = \frac{q_{xe} + q_{xw}}{2}, \quad q_{xe} = h_P \max(u_e, 0) - h_E \max(-u_e, 0),$$

$$q_{xw} = h_W \max(u_w, 0) - h_P \max(-u_w, 0);$$

$$q_{yne} = \frac{q_{yn} + q_{ys}}{2}; \quad q_{yn} = h_P \max(v_n, 0) - h_N \max(-v_n, 0),$$

$$q_{ys} = h_S \max(v_s, 0) - h_P \max(-v_s, 0).$$

При вычислении потоков — произведений $q_{xE}u_e, q_{yne}u_{ne}, q_{xEn}v_{en}, q_{yN}v_N$ также используются аппроксимации ван Лира [14] с ограничителем [15], рассмотренные выше. Для устойчивого по времени интегрирования уравнений (4)–(6) величина шага по времени выбирается из условия

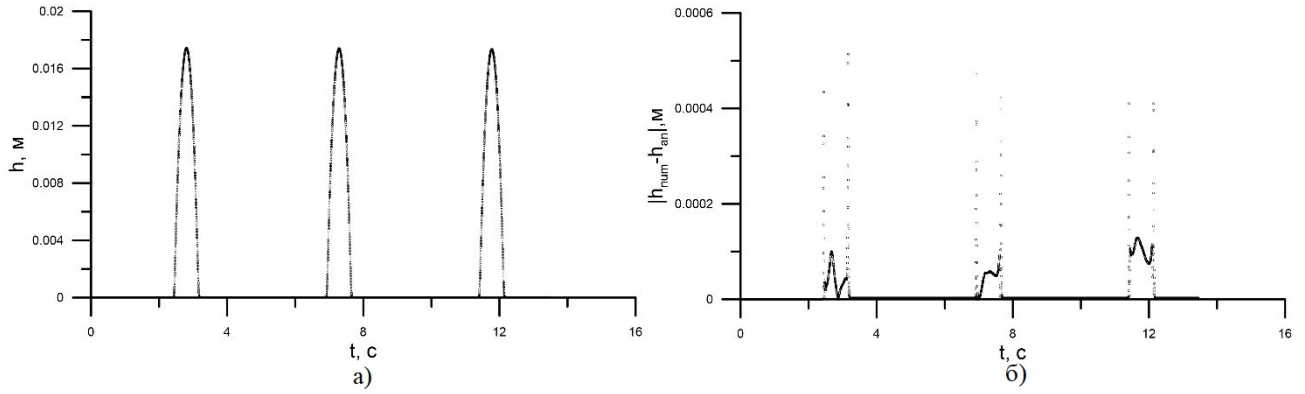


Рис. 2. Сравнение расчетов (открытые кружки) и аналитического решения (линия) для глубины жидкости h (а) и погрешность расчетов (б) для 2D задачи Такера [16] в точке (1.004м; 1.004м) для трех периодов колебаний жидкости

$$\Delta t_k < \frac{\max(\Delta x, \Delta y)}{2 \left(|\vec{w}_P^k| + \sqrt{gh_P^k} \right)}. \quad (7)$$

Такой шаг обеспечивает неотрицательность глубины потока ($h_P^{k+1} \geq 0$).

В качестве задачи, на которой проводилось тестирование точности вычислений и качества распараллеливания, рассматривалась классическая двумерная задача Такера о колебании жидкости без трения в водоеме, дно которого представляет собой параболоид. Аналитическое решение задачи представлено в [16] и имеет вид:

$$\begin{cases} h(x,y,t) = \frac{\eta h_0}{a^2} [2(x - L/2)\cos(\omega t) + 2(y - L/2)\sin(\omega t) - \eta] - z_b(x,y); \\ u(x,y,t) = -\eta\omega \times \sin(\omega t); \\ v(x,y,t) = \eta\omega \times \cos(\omega t), \end{cases} \quad (8)$$

а рельеф дна определяется формулой:

$$z_b(x,y) = -h_0 \left(1 - \frac{r(x,y)^2}{a^2} \right), \quad r^2(x,y) = (x - L/2)^2 + (y - L/2)^2.$$

Здесь $L_x = L_y = L = 4\text{м}$, $a = 1\text{м}$, $h_0 = 0.1\text{м}$, $\eta = 0.5\text{м}$, $\omega = \sqrt{2gh_0}/a$, c^{-1} .

Для этого случая движущаяся береговая линия представляет собой эллипс. Свободная поверхность жидкости совершает периодическое движение и остается плоской во времени. Чтобы представить себе этот случай, нужно рассмотреть сосуд в форме параболоида, внутри которого вращается жидкость. В расчетах использовалась сетка 500×500 . По времени вычислительный процесс длился три периода колебаний жидкости в параболоиде. Шаг по времени определялся из условия устойчивости разностных схем (7).

На рис. 2, а, представлены графики изменения глубины потока с течением времени в точке ($x = 1,004\text{м}$; $y = 1,004\text{м}$). Из рисунка видно, что в рассматриваемом промежутке времени волна троекратно набегаёт на указанное положение на поверхности параболоида. Открытые значки на рис. 2, а, соответствуют результатам численных расчетов, а линия — аналитическому решению [16]. На рис. 2, б, в расчетные моменты времени изображен модуль разности численного и аналитического решений для глубины течения в этой же

точке. Видно, что наибольшее расхождение для рассматриваемого положения имеет место в случае начала и конца прохождения волны. Последнее может быть связано с тем, что в аналитическом решении границы волны являются непрерывными функциями от времени и координат, а в численных расчетах положение границы меняется дискретно с шагом, равным размеру ячейки. Тем не менее, когда жидкость накрывает рассматриваемую точку, погрешность расчета глубины уменьшается в 4 раза.

3. Параллельная реализация. Построенная выше полунейвная разностная схема на разнесенных разностных сетках имеет большой потенциал для создания эффективной параллельной программы, поскольку на каждом шаге по времени вычисления глубины или компонент скорости по формулам (4)–(6) могут производиться одновременно и независимо в каждом вычислительном узле сетки.

Остановимся подробнее на программной реализации этой разностной схемы. На рис. 3 приведена блок-схема программы, выполняющей вычисления по описанным выше разностным схемам. Основной цикл по времени включает блоки: а) расчета потоков G_e, G_w, G_n, G_s через грани конечных объемов и вычисление глубины h по (4) и уровня жидкости $\xi = h + z_b$; б) расчета потоков для уравнения (5) и вычисление компоненты скорости u_e^{k+1} по (5); в) расчета потоков для уравнения (6) и вычисление компоненты скорости v_n^{k+1} по (6); г) обновление полей на предыдущем шаге по времени.

В каждом из блоков а)–г) выполняется один или два вложенных цикла по сеточной области, в которых по явным формулам производятся вычисления. Например, основные блоки программы — блок а), в котором производится расчет глубины потока и уровня жидкости, и блок г), в котором выполняется обновление полей для следующего шага по времени, можно представить в следующем виде:

```

while (time < TimeT) { // основной цикл по времени
...
for (int i=1; i < Nx; i++) {
  for (int j=1; j < Ny; j++) {
// расчет потоков для уравнения (4)
    Ge = MUSCL(h[i-1][j], h[i][j], h[i+1][j], h[i+2][j], u[i+1][j]);
    Gw = MUSCL(h[i-2][j], h[i-1][j], h[i][j], h[i+1][j], u[i][j]);
    Gn = MUSCL(h[i][j-1], h[i][j], h[i][j+1], h[i][j+2], v[i][j+1]);
    Gs = MUSCL(h[i][j-2], h[i][j-1], h[i][j], h[i][j+1], v[i][j]);
// расчет глубины потока и уровня жидкости по явным формулам
    hnext[i][j] = h[i][j] - dt/dx*(Ge-Gw) - dt/dy*(Gn-Gs);
    dzetanext[i][j] = hnext[i][j] + zb[i][j];
  } // end for
// end for
...
// обмен значениями
for (int i=0; i < Nx+1; i++)
  for (int j=0; j < Ny+1; j++) {
    h[i][j] = hnext[i][j];
    u[i][j] = unext[i][j];
    v[i][j] = vnext[i][j];
  } // end for
time = time + dt;
} // end while

```

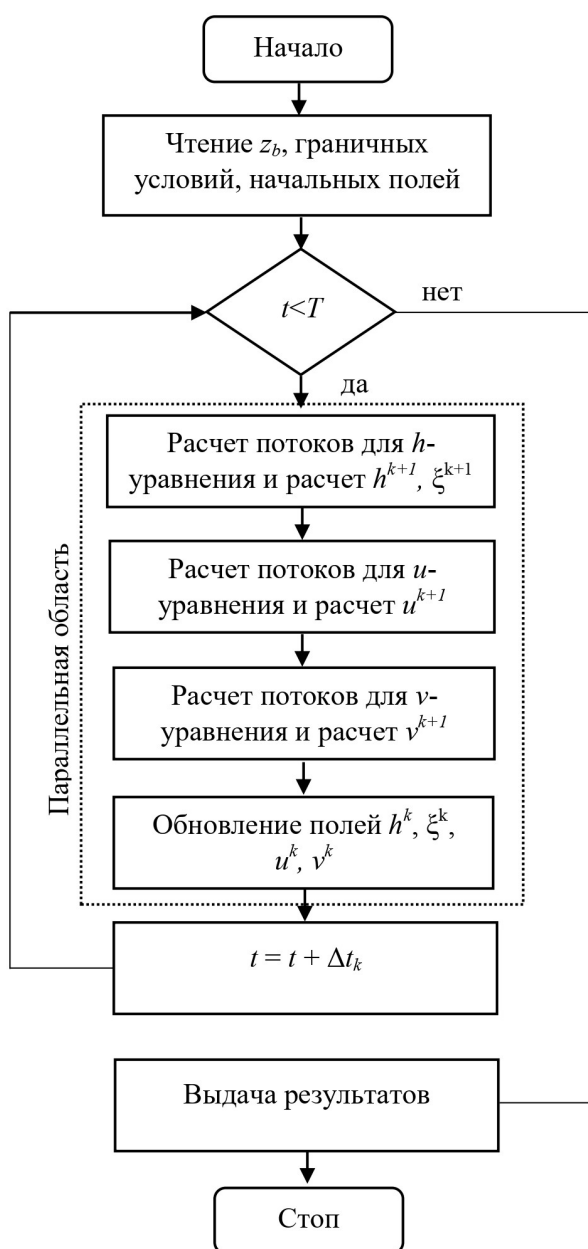


Рис. 3. Блок-схема вычислительного процесса с указанием области параллельных вычислений

Из этого фрагмента программы видно, что программа состоит из последовательности блоков двойных вложенных циклов, в которых активно производится использование и обновление одних и тех же относительно небольших по величине двумерных статических массивов. Все вещественные переменные и массивы в программе были отнесены к типу `double`. Расчеты проводились на сервере с общей памятью объемом 192Гб и двумя двенадцатядерными процессорами Intel Xeon Silver 4214 2,2 ГГц и одной графической картой GPU NVIDIA Geforce RTX2080 Ti. Расчет по последовательной программе для рассматриваемых условий проведения вычислений на сервере составил 547,7 с (сетка 500×500) и 4606,4 с (сетка 1000×1000).

Построение параллельных программ было выполнено с помощью мало затратных в параллельном программировании технологий OpenMP (Open MultiProcessing) и OpenACC (Open Accelerators), предназначенных, в основном, для многоядерных вычислительных систем CPU+GPU с общей памятью. На рис. 3 штриховой линией выделена часть блок-схемы, в которой производились параллельные вычисления. Внутри этой области выполнялось распараллеливание вложенных циклов (см. приведенный фрагмент программы) средствами параллельных технологий OpenMP или OpenACC, т.е. выполнение итераций вложенных циклов равномерно распределялось между активными нитями/процессорами ядер центрального или графического процессоров. Ниже представлен фрагмент OpenMP-программы, в которую перед каждым вложенным циклом добавлялись директивы `#pragma omp parallel for`:

```

while (time<TimeT) { // основной цикл по времени
...
#pragma omp parallel for collapse(2) private(Ge, Gw, Gn, Gs)
// распределение итераций между ядрами CPU
for (int i=1; i<Nx; i++) {
  for (int j=1; j<Ny; j++) {
    Ge = ...; // расчет потоков для уравнения (4)
    ...
    hnext[i][j] = h[i][j] - dt/dx*(Ge-Gw) - dt/dy*(Gn-Gs);
    // расчет глубины потока
    dzetanext[i][j] = hnext[i][j]+zb[i][j];
  } // end for
} // end for
...
} //end while

```

При компиляции всех программ использовался компилятор Portland Group 21.9 pgcc, который позволяет создавать параллельные программы с использованием технологий OpenMP и OpenACC для многоядерных систем с центральными и графическими процессорами.

На рис. 4 представлены графики ускорения и эффективности параллельной программы, созданной с помощью OpenMP технологии для расчетов на сетках 500×500 и 1000×1000 узлов. Ускорение S_p рассчитывалось как отношение измеренного времени счета при запуске параллельной программы на одной нити/ядре ($p = 1$) к измеренному времени счета параллельной программы на p нитях/ядрах. Рассматривались значения $p = 1, 3, 6, 12, 24$. Эффективность параллельной программы оценивалась как отношение ускорения к числу используемых нитей/ядер.

Из рис. 4 видно, что использование технологии OpenMP позволяет более чем в 15 раз ускорить вычислительный процесс. Причем программа хорошо масштабируется: при кратном увеличении вычислительной работы (\sim в 8 раз) ускорение и эффективность программы практически не меняется. Заметим, однако, что при увеличении количества используемых ядер центральных процессоров темп увеличения ускорения и значения эффективности параллельной программы снижаются. Последнее связано с увеличением нагрузки на оперативную память, точнее, с уменьшением пропускной способности памяти в расчете на одно активное ядро при использовании всех доступных физических ядер центрального процессора. Скорость выполнения арифметических операций в программе зависит

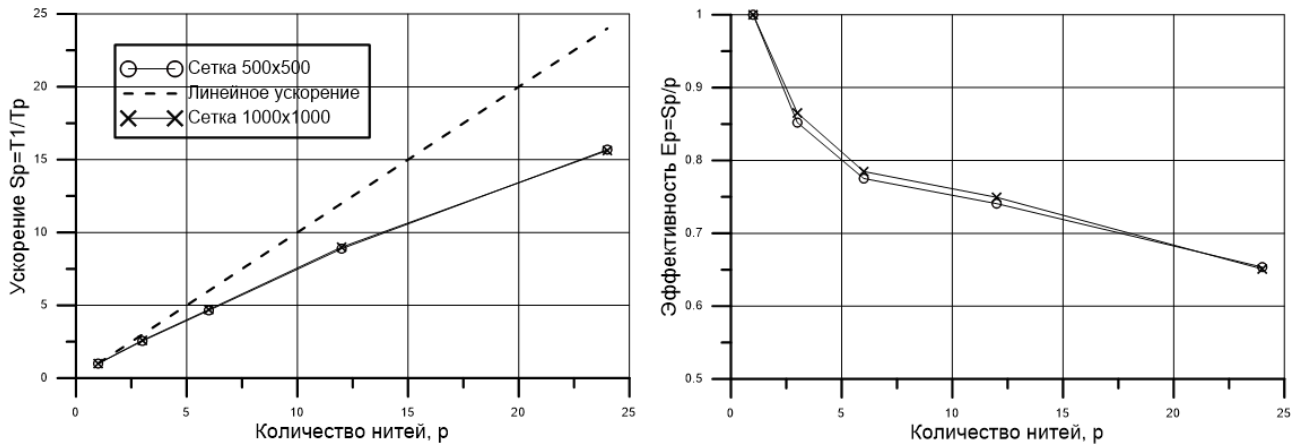


Рис. 4. Ускорение и эффективность параллельной OpenMP-программы. Штриховая линия — график зависимости идеального ускорения $S_p = p$

не только от фактической скорости процессора и количества его ядер, но и от времени выполнения операций с памятью.

Ниже представлен фрагмент OpenACC-программы, в которую перед каждым вложенным циклом добавлялись директивы `#pragma acc kernels`. Кроме того, перед началом вычислений было выполнено копирование/создание всех используемых в расчетах на графическом процессоре массивов.

```
// копирование данных из ОЗУ компьютера
// в память графической карты и обратно
#pragma acc data copy(hnext, unext, vnext, dzetanext, h, u, v)
#pragma acc data copyin(zb)
#pragma acc data create(u, v, hcх, hcу, qх, qу, qcх, qcy)
while (time < TimeT) { // основной цикл по времени
...
#pragma acc kernels // распределение итераций цикла между ядрами GPU
for (int i=1; i<Nx; i++) {
  for (int j=1; j<Ny; j++) {
    Ge = ...; // расчет потоков для уравнения (4)
    ...
    hnext[i][j] = h[i][j] - dt/dx*(Ge-Gw) - dt/dy*(Gn-Gs);
    dzetanext[i][j] = hnext[i][j] + zb[i][j];
  } // end for
} // end for
...
} // end while
```

Использование технологии OpenACC для программирования на графической карте на рассматриваемой вычислительной системе и при тех же условиях проведения расчетов дает ускорение более чем в 25 (21,3 с против 547,7 с) на сетке 500×500 и в 31 раз (145,4 с против 4606,4 с) на сетке 1000×1000 .

Заключение. Для численного моделирования нестационарных изотермических турбулентных течений в речных потоках сформулирована математическая модель, опирающаяся на приближение мелкой воды для уравнений Рейнольдса для несжимаемой

жидкости, эффективный численный метод, обеспечивающий в рамках использования метода конечного объема, структурированных разнесенных сеток и полунявных разностных схем выполнения на разностном уровне законов сохранения массы и импульса. Вычислительная реализация предложенной модели и метода была протестирована на известном аналитическом решении и распараллелена с помощью технологий OpenMP и OpenACC на гибридной многоядерной системе с общей памятью. Расчеты показали, что использование технологии OpenMP для двух двенадцатиядерных центральных процессоров позволяет более чем в 15 раз ускорить вычислительный процесс. Использование технологии OpenACC при расчетах на этой же многоядерной системе и графическом процессоре NVIDIA GeForce RTX2080 Ti дает ускорение более чем в 25 раз.

Список литературы

1. Saint-Venant A. J. B. Theorie du Mouvement non permanent des Eaux // Institut de France, Acad. des Sci. de Paris. 1871. 73 (3) 147; 73 (4) 237.
2. Gusev O. I., Khakimzyanov G. S., Skiba V. S., Chubarov L. B. Shallow water modeling of wave-structure interaction over irregular bottom // Ocean Engineering. 2023. V. 267. Art. 113284.
3. Brand S. Parallel algorithm for numerical solution of the shallow water equation // Proceedings of the Czech-Japanese Seminar in Applied Mathematics. 2006. Czech Technical University in Prague, September 14–17, 2006. P. 25–36.
4. Габдулина М. И. Параллельный алгоритм численного решения двухслойной модели мелкой воды в двумерном случае // Суперкомпьютерные дни в России 2016 // Russian Supercomputing Days 2016 // RussianSCDays.org. С. 460–467.
5. Чаплыгин А. В., Гусев А. В. Гибридная модель мелкой воды с использованием технологий MPI-OpenMP // Проблемы информатики, 2021. № 1. С. 65–82.
6. Juliati S., Gunawan P. H. OpenMP architecture to simulate 2D water oscillation on paraboloid // 5th International Conference on Information and Communication Technology (ICoIC7), 2017. P. 1–5.
7. Liu Q., Qin Y., Li G. Fast Simulation of Large-Scale Floods Based on GPU Parallel Computing // Water. 2018. V. 10(5):589.
8. Zhang S., Li W., Jing Zh, Yi Y., Zhao Y. Comparison of Three Different Parallel Computation Methods for a Two-Dimensional Dam-Break Model // Mathematical Problems in Engineering. 2017. V. 2017, Article ID 1970628, 12 pages.
9. Arnoldy A., Adytia D. Performance of Staggered Grid Implementation of 2D Shallow Water Equations using CUDA Architecture // 2019 12th International Conference on Information & Communication Technology and System (ICTS), Surabaya, Indonesia, 2019, P. 286–290.
10. Churuksaeva V., Starchenko A. Mathematical modeling of a river stream based on a shallow water approach // Procedia Computer Science. 2015. V. 66. P. 200–209.
11. Rastogi A. K., Rodi W. Predictions of heat and mass transfer on open channels // ASCE Journal of the Hydraulics Division. 1978. V. 104(3). P. 397–420.
12. Launder B. E., Spalding D. B. The numerical computation of turbulent flows // Computer Methods in Applied Mechanics and Engineering. 1974. V. 2(3). P. 269–289.
13. Stelling G. S., Duimijer S. P. A. A staggered conservative scheme for every Froude number in rapidly varied shallow water flows // Int. Journal for Numerical Methods in Fluids. 2003. V. 43. P. 1329–1354.
14. van Leer B. Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method // Journal of Computational Physics. 1979. V. 32(1). P. 101–136.
15. Cada M., Torrilhon M. Compact third-order limiter functions for finite volume methods // Journal of Computational Physics. 2009. V. 228. P. 4118–4145.

16. Thacker W. C. Some exact solutions to the nonlinear shallow water wave equations // Journal of Fluid Mechanics. 1981. V. 107. P. 499–508.



Старченко Александр Васильевич — E-mail: starch@math.tsu.ru. Тел.: +7 (382) 252-9553. Заведующий кафедрой вычислительной математики и компьютерного моделирования Национального исследовательского Томского

государственного университета, ведущий научный сотрудник лаборатории численного моделирования и высокопроизводительных ресурсов Томского филиала Федерального исследовательского центра информационных и вычислительных технологий. Область научных интересов: численные методы, математическое мо-

делирование, высокопроизводительные вычисления.

Starchenko Alexander Vasilievich — E-mail: starch@math.tsu.ru. Tel.: +7 (382) 252-9553. Doctor of Physical and Mathematical Sciences, Professor, Head of Department of Computational Mathematics and Computer Modelling of National Research Tomsk State University, Leading Researcher at the Laboratory of Numerical Modeling and High-Performance Resources of the Tomsk Branch of the Federal Research Center for Informational and Computational Technologies. Area of scientific interests: numerical methods, mathematical modeling, high-performance computing.

Дата поступления — 31.01.2024