DOI: 10.7242/1999-6691/2016.9.3.25
УДК 532.5

ИССЛЕДОВАНИЕ ЭФФЕКТИВНОСТИ ПАРАЛЛЕЛЬНОЙ РЕАЛИЗАЦИИ АЛГОРИТМА SIMPLE НА МНОГОПРОЦЕССОРНЫХ ЭВМ

С.В. Лашкин, А.С. Козелков, А.В. Ялозо, В.Ю. Герасимов, Д.К. Зеленский

Государственная корпорация по атомной энергии «Росатом» ФГУП «РФЯЦ - ВНИИЭФ», Саров, Нижегородская обл., Российская Федерация
Нижегородский государственный технический университет им. Р.Е. Алексеева, Нижний Новгород, Российская Федерация

Обсуждаются особенности параллельной реализации алгоритма SIMPLE численного решения системы уравнений Навье–Стокса на произвольных неструктурированных сетках. Применяются итерационные схемы последовательного и параллельного вариантов алгоритма. При решении с помощью параллельного варианта SIMPLE особое внимание акцентируется на межпроцессных обменах расчетными данными при условии декомпозиции сеточной модели с созданием фиктивных ячеек. Подробно рассмотрены особенности хранения распределенных матриц, а также выполнение матрично-векторных операций в параллельном режиме. Показано, что предложенный способ позволяет уменьшить число межпроцессных обменов. На серии численных экспериментов продемонстрировано влияние настроек многосеточного решателя СЛАУ на общую эффективность алгоритма, а именно: типы используемых циклов – V, W, или F, количество итераций сглаживателя и количество ячеек для огрубления. Приводятся два способа оценки эффективности распараллеливания численного алгоритма: прямой и косвенный. Описывается решение задач внутреннего турбулентного течения жидкости в трубе круглого сечения и в канале за обратным уступом, а также приводится решение задачи внешнего обтекания потоком воздуха препятствия «Ahmed body», с оценкой эффективности распараллеливания по двум алгоритмам. Установлено, что предложенный параллельный вариант алгоритма SIMPLE предоставляет возможность эффективно считать задачи на тысяче процессов. На основе полученных результатов даются общие рекомендации по оптимальному выбору как настроек многосеточного решателя, так количества ячеек на одном процессе.

Ключевые слова: вычислительная гидродинамика, алгоритм SIMPLE, многосеточный решатель, моделирование

EFFICIENCY ANALYSIS OF PARALLEL IMPLEMENTATION OF SIMPLE ALGORITHM ON MULTI-PROCESSOR COMPUTERS

S.V. Lashkin, A.S.Kozelkov, A.V. Yalozo, V.Yu. Gerasimov and D.K. Zelensky

Public Corporation to Atomic Power Engineering "Rosatom" FSUE "RFNC-VNIIEF", Nizhny Novgorod Region, Sarov, Russian Federation
Nizhny Novgorod State Technical University n.a. R.E. Alekseev, Nizhny Novgorod, Russian Federation

This paper describes the details of parallel implementation of a SIMPLE algorithm for numerical solution of the Navier–Stokes system of equations on arbitrary unstructured grids. Implemented iteration schemes of serial and parallel options of the SIMPLE algorithm are shown. When describing parallel implementation, special attention is paid to the description of computational data exchange between the processors under the condition of the grid model decomposition using fictitious cells. We discuss specific features for the distributed matrices storage and implementation of the vector-matrix operations in the parallel mode. We show that the suggested way of matrix storage will allow reducing the number of inter-processor exchange. A series of numerical experiments illustrates multi-grid SLAE solver tuning as it affects the general efficiency of the algorithm (it includes the types of the cycles used – V, W and F, the number of smoothing operator iterations, and the number of cells for coarsening). Two ways (direct and indirect) of the efficiency evaluation for the numerical algorithm parallelization are shown. The paper provides the results of the internal and external flow problem solution with parallelization efficiency evaluation by two algorithms. It is shown that the suggested parallel implementation makes it possible to do efficient computation on the problems on a thousand of processors. Based on the results produced, general recommendations are given on the choice of optimal tuning of the multi-grid solver and optimal number of cells per processor.

Key words: computational fluid dynamics (CFD), SIMPLE algorithm, multi-grid solver, modeling

1. Введение

Течения жидкостей и газов описываются краевыми задачами для системы уравнений Навье–Стокса, представляющих собой нелинейные дифференциальные уравнения в частных производных [1, 2]. Проблемы дискретизации этих уравнений, а также их численного решения составляют один из ключевых этапов математического моделирования. Одним из широко известных, практикуемых с этой целью способов является алгоритм SIMPLE (Semi-Implicit Method for Pressure-Linked Equations) [3], основанный на использовании полунеявной схемы с расщеплением по физическим процессам и имеющий несколько модификаций [4]. Основной класс задач, для реализации которых подходит данный алгоритм, — это процессы течения вязкой несжимаемой и слабосжимаемой жидкости. Также существует возможность обобщения алгоритма SIMPLE на случай сильносжимаемых вязких течений с произвольными числами Маха [1, 5, 6]. Данному алгоритму посвящено много статей и монографий, в которых подробно описываются принципы его реализации на произвольных неструктурированных сетках с применением различных схем и способов дискретизации на неортогональных (в основном тетраэдральных) сетках,

позволяющих получать результаты с достаточной степенью соответствия экспериментальным данным [7, 8]. Большинство опубликованных работ основывается на последовательной версии алгоритма SIMPLE, а его параллельная составляющая отражена недостаточно.

В последние годы наряду с проблемами численной реализации, касающимися точности и расчета на неортогональных сетках, все более важной становится эффективность осуществления параллельных вычислений на многопроцессорных суперЭВМ, содержащих десятки тысяч процессорных ядер. Необходимость расчетов на подробных сетках (порядка сотен миллионов ячеек), например, при исследовании различных энергетических установок со сложными физическими процессами, требует использования LES моделей и, следовательно, мощных ЭВМ, поскольку решение задач на малопроцессорных машинах (с числом процессорных ядер порядка ста) может длиться сотни часов. Сокращение времени счета достигается двумя путями.

Первый путь — это увеличение производительности расчетов, например, благодаря сопроцессорорам [9, 10], что ведет к большим затратам, связанным с эффективной адаптацией неявных алгоритмов на нерегулярных сетках к архитектуре программного обеспечения сопроцессоров и не гарантирует положительный результат. Дело в том, что подобные CFD-приложения (приложения к вычислительной гидродинамике — Computational Fluid Dynamics) [11] относятся к классу «memory-bound», и скорость нерегулярной выборки данных из оперативной памяти является главным ограничителем их производительности, к тому же сопроцессоры не превосходят многоядерные процессоры по показателю «отношение скорости выборки данных к скорости вычислений», особенно в случае нерегулярной (произвольной) выборки [12]. Другими словами, «memory-bound»-приложения могут потратить большую часть своего времени на обработку передвижения по иерархической структуре памяти, и, поскольку время операции с памятью примерно от десяти до ста раз больше по сравнению со временем арифметической операции с плавающей запятой, выигрыш от использования сопроцессоров для CFD-приложений будет минимальным. Эта особенность значительно сокращает скорость выполнения основных операций с разреженной матрицей нерегулярной структуры (приближенного разложения, умножения матрицы на вектор и другого), так как доступ к данным происходит нерегулярно. Такой доступ к памяти влечет за собой увеличение вспомогательных (неарифметических) операций, и хотя некоторые разработчики получают ускорение счета [13, 14], но оно гораздо «скромнее» отношения пиковой производительности сопроцессоров к пиковой производительности процессоров. Сопроцессоры эффективны в приложениях, реализующих явные схемы интегрирования в газовой динамике [15], молекулярной динамике [16] и методе Монте-Карло [17]. Здесь один сопроцессор ускоряет счет до 5 раз по сравнению с одним многоядерным процессором. Если же прибегать к блочно-структурированным сеткам, то можно достигнуть ускорения до 15 раз. В случае неструктурированных сеток максимальное ускорение не превышает 2 раз [13–15]. По этой причине сопроцессоры и графические ускорители еще не получили широкого распространения в CFD-кодах, хотя на ближайшие 5÷10 лет эта технология предполагается предпочтительной для построения суперкомпьютеров. Для проведения масштабных гидродинамических расчетов потребуется несколько десятков или даже сотен таких ускорителей, что сразу же приведет к необходимости брать на вооружение классические схемы распараллеливания с применением MPI-интерфейсов. Поэтому, наряду с разработкой алгоритмов для сопроцессоров, исследования по оптимизации существующих и разработка новых алгоритмов распараллеливания на базе MPI остаются актуальными.

Второй путь включает эффективное выполнение алгоритмов на основе декомпозиции расчетной модели с фиктивными ячейками, позволяющей распараллелить действия и минимизировать межпроцессные обмены [18]. При этом подразумевается, что известна схема итерационного алгоритма и «места» межпроцессных обменов, что исключает избыточную передачу данных от процесса к процессу, которые существенно снижают эффективность алгоритма и в итоге увеличивают общее время счета задачи.

Необходимо отметить, что последние тенденции ускорения параллельных вычислений связаны в первую очередь с архитектурами нового поколения [9, 10] и оптимизацией программ к многопоточным с общей памятью [19]. В данной статье рассматривается итерационная схема алгоритма SIMPLE и его реализация на высокопараллельных ЭВМ, включающая декомпозицию расчетной модели с использованием фиктивных ячеек, работу параллельного многосеточного решателя СЛАУ с его параметрами, структурой и операциями с матрицей. Приводятся результаты численных экспериментов и оценка общей эффективности алгоритма SIMPLE, рассчитанная как аналитически, так и с помощью специальных программ.

2. Итерационная схема алгоритма SIMPLE

Если при разработке численного метода решения уравнений Навье–Стокса ориентироваться на произвольные неструктурированные сеточные модели, состоящие из ячеек произвольной формы, то наиболее оптимальным для дискретизации является метод конечных объемов [1]. Именно этот метод выбран в качестве основного для решения задач вычислительной аэрогидродинамики в пакете программ ЛОГОС [18, 20–23], предназначенном для решения связанных и сопряженных задач тепломассопереноса и прочности на параллельных ЭВМ. Пакет программ ЛОГОС успешно прошел верификацию и показал

достаточно хорошие результаты на серии различных гидродинамических задач [23], в том числе с учетом турбулентных и нестационарных течений [20, 42–44], а также геофизических явлений [45, 46].

Один из численных алгоритмов, реализованных в пакете программ ЛОГОС, — это полунеявный итерационный алгоритм SIMPLE [1–3], предназначенный в первую очередь для решения задач несжимаемых и слабосжимаемых течений с возможностью расширения на транс- и сверхзвуковые течения с произвольными числами Маха [5].

Несжимаемые течения, то есть течения с постоянной плотностью, описываются, как правило, следующей системой уравнений Навье–Стокса:

$$\begin{cases} \rho\nabla\cdot\mathbf{u}=0, \\ \rho\dfrac{\partial\mathbf{u}}{\partial t}+\rho\nabla\cdot\left(\mathbf{u}\otimes\mathbf{u}\right)-\nabla\cdot\boldsymbol{\tau}=-\nabla p. \end{cases} \tag{1}$$

В (1) используются общепринятые обозначения: ρ — плотность; $\mathbf{u}$ — вектор скорости осредненного течения в трехмерной системе координат; p — давление; t — время; $\boldsymbol{\tau}=\boldsymbol{\tau}_{\mu}+\boldsymbol{\tau}_{t}$ — сумма молекулярной и турбулентной составляющих вязкой части тензора напряжений соответственно.

В случае турбулентных течений, осредненных по Рейнольдсу, вычисление компонент тензора с учетом гипотезы Буссинеска производится по формулам:

$$\begin{aligned} \tau_{\mu}&=\mu\left(\nabla\mathbf{u}+\nabla\mathbf{u}^{\mathrm{T}}-\frac{2}{3}\mathbf{I}\left(\nabla\cdot\mathbf{u}\right)\right), \\ \tau_{t}&=\mu_{t}\left(\nabla\mathbf{u}+\nabla\mathbf{u}^{\mathrm{T}}-\frac{2}{3}\mathbf{I}\left(\nabla\cdot\mathbf{u}\right)\right)-\frac{2}{3}\rho k\mathbf{I}, \end{aligned} \tag{2}$$

где μ — молекулярная вязкость, μ_t — турбулентная вязкость, k — кинетическая энергия турбулентности, $\mathbf{I}$ — единичный тензор второго ранга, «T» — символ операции транспонирования. Подробно слагаемые, входящие в равенства (2), а также особенности моделирования турбулентных течений представлены в [24].

Решение системы уравнений (1) «напрямую» невозможно, так как в приближении несжимаемой среды отсутствует связь поля скорости с полем давления. Изначально давление не входит в уравнение неразрывности, и попытка одновременного решения двух уравнений без привлечения специальных алгоритмов приводит к неразрешимым трудностям. Существующие подходы не универсальны и накладывают те или иные ограничения. Например, метод искусственной сжимаемости [25] не эффективен при расчете течений вдоль длинных каналов или в сложных системах вентиляции, метод функций тока и переноса завихренности [2] нельзя обобщить на случай трехмерных течений и так далее.

Одним из универсальных и доминирующим на сегодняшний день подходом, обеспечивающим связь полей скорости и давления, является алгоритм SIMPLE. Его суть заключается в расщеплении исходных величин и последующем их согласовании на каждом итерационном шаге:

$$\begin{aligned} \mathbf{u}^{n+1}&=\mathbf{u}^{*}+\mathbf{u}', \\ p^{n+1}&=p^{n}+p', \end{aligned}$$

где $\mathbf{u}'$ и p' — поля приращений скорости и давления соответственно, n — решение в предыдущий момент времени или с предыдущей итерации, $\mathbf{u}^{*}$ — предварительное поле скорости. Такое расщепление, как будет показано далее, позволяет решать уравнение неразрывности относительно приращения давления, что невозможно сделать в исходных уравнениях (1).

Запишем систему дифференциальных уравнений (1) в дискретном виде:

$$\begin{cases} \rho\nabla\cdot u^{n+1}=0, & 1 \\ \rho\dfrac{\partial\left(\mathbf{u}^{n+1}-\mathbf{u}^{n}\right)}{\partial t}+\rho\nabla\cdot\left(\mathbf{u}^{n}\otimes\mathbf{u}^{n+1}\right)-\nabla\cdot\boldsymbol{\tau}^{n+1}=-\nabla p^{n+1}. & 2 \end{cases} \tag{3}$$

Согласно общепринятой структуре алгоритма SIMPLE на первом этапе рассчитывается предварительное поле скорости $\mathbf{u}^{*}$ при поле давления с предыдущего итерационного шага, то есть в уравнении движения скорость $\mathbf{u}^{n+1}$ заменяется на $\mathbf{u}^{*}$:

$$\rho\frac{\mathbf{u}^{*}-\mathbf{u}^{n}}{\Delta t}+\rho\nabla\cdot\left(\mathbf{u}^{n}\otimes\mathbf{u}^{*}\right)-\nabla\cdot\boldsymbol{\tau}^{n+1}=-\nabla p^{n}. \tag{4}$$

Формально решение данного уравнения возможно с любым полем давления, но найденное предварительное поле скорости не будет удовлетворять уравнению неразрывности. Поэтому на втором этапе вычисляется значение приращения скорости, выражение для которого получается путем вычитания уравнения (4) из уравнения $(3)_2$ и отбрасывания второго и третьего слагаемых:

$$u' = -\frac{\Delta t}{\rho}\nabla p' . \tag{5}$$

Видно, что поправка для скорости зависит от величины градиента приращения давления. Само же уравнение для поправки давления следует из уравнения неразрывности после подстановки в него равенства (5):

$$\rho\nabla\cdot\mathbf{u}^* = \nabla\cdot\left(\Delta t\nabla p'\right). \tag{6}$$

Реализовать представленные выше действия можно в виде «полунеявной» итерационной процедуры алгоритма SIMPLE [1–3] Она включает следующие шаги:

1. Вычисление градиентов трех компонент скоростей и давления.
2. Решение уравнения сохранения количества движения (4) для определения предварительного поля скорости $\mathbf{u}^*$, не удовлетворяющего уравнению неразрывности.
3. Нахождение массовых потоков на гранях контрольного объема: $m_f^* = \rho S_f\left(\mathbf{u}_f^*\cdot\mathbf{n}_f\right)$, где f — грань контрольного объема, S_f — площадь грани и $\mathbf{n}_f$ — единичный вектор нормали.
4. Расчет поля приращения давления на основании уравнения (6).
5. Установление градиента приращения давления и корректировка поля скорости по уравнению (5) и перерасчет массового потока на гранях $m_f^* = \rho S_f\left(\mathbf{u}_f^{n+1}\cdot\mathbf{n}_f\right)$, который удовлетворяет исходному уравнению неразрывности.
6. Проверка условия выхода (например, по величине невязки расчетных полей) и, при необходимости, возврат к шагу 1.

В результате применения данной процедуры получаются четыре системы линейных алгебраических уравнений (СЛАУ): три системы для трех компонент скоростей и одна для приращения давления. При решении дополнительных уравнений, например, уравнений теплопроводности или уравнений, моделирующих турбулентность, соответственно увеличивается число СЛАУ.

3. Описание параллельной версии вычислительной процедуры SIMPLE

Использование метода контрольных объемов подразумевает этап восстановления всех физических величин, участвующих в расчете (скорости, давления, температуры и другого), а также массового потока на гранях, разделяющих контрольные объемы. Для вычисления значений на гранях применяются различные схемы дискретизации, точность которых ограничена лишь собственными возможностями и устойчивостью. Например, известная линейная схема второго порядка точности [1] для определения произвольной величины φ_f на смежной грани контрольных объемов (Рис. 1) записывается в виде:

$$\varphi_f = \lambda\varphi_P + (1-\lambda)\varphi_N , \tag{7}$$

где $\lambda = \dfrac{\left|\mathbf{n}_f\cdot\mathbf{d}_{Nf}\right|}{\left|\mathbf{n}_f\cdot\mathbf{d}_{Nf}\right| + \left|\mathbf{n}_f\cdot\mathbf{d}_{Pf}\right|}$ — геометрический интерполяционный фактор.

Понятно, что употребление схемы второго порядка точности требует знания значений φ_P и φ_N в контрольных объемах P и N. Установить φ_f в последовательном режиме не составляет труда. В параллельном же случае, при совпадении грани с границей раздела MPI-процессов, значение φ_f найти невозможно, поскольку образующие грань ячейки P и N физически принадлежат разным процессам (Рис. 1). Существует два варианта решения данной проблемы. Первый — введение понятия «обменных»

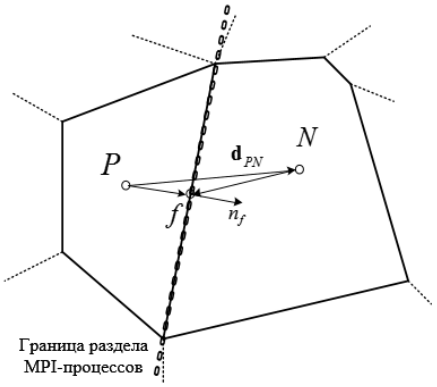

Рис. 1. Два соседних контрольных объема P и N; f – смежная грань, d_{PN} – вектор, соединяющий центры объемов, d_{Pf} и d_{Nf} – вектора, соединяющие центры объемов P и N с центром грани f, $\mathbf{n}_f$ – вектор нормали к смежной грани

граней f' и их «специальная» обработка в расчетном алгоритме. Однако это приводит к дополнительным сложностям, так как требуется написание отдельных (по сравнению с последовательным вариантом) циклов обработки. Например, чтобы вычислить значение упомянутого выше φ_f, необходимо вначале сохранить значение смежной ячейки в обменной грани f', затем произвести межпроцессный обмен между фиктивными гранями соседних MPI-процессов, и только потом рассчитать величину φ_f на расчетной грани f, причем такая последовательность необходима для каждой из искомых величин. Но наиболее существенным недостатком при этом является невозможность расширения сеточного шаблона с целью использования его в дальнейшем для построения схем третьего и выше порядков точности, так как информация об оппозитных ячейках попросту недоступна. Подобная схема используется в [26]. Второй вариант учета перехода от одного MPI-процесса к другому заключается в следующем — вводится слой фиктивных ячеек P' и N' таким образом, что они становятся прообразами счетных ячеек соседних MPI-процессов, и организуются MPI-обмены между фиктивными и счетными ячейками (Рис. 2). В этом случае, согласно равенству (7), выражения для величины φ_f на разных процессах запишутся в виде (Рис. 2):

$$(\varphi_f)_{1\text{процесс}} = \lambda\varphi_P + (1-\lambda)\varphi_{N'},$$
$$(\varphi_f)_{2\text{процесс}} = \lambda\varphi_{P'} + (1-\lambda)\varphi_N.$$

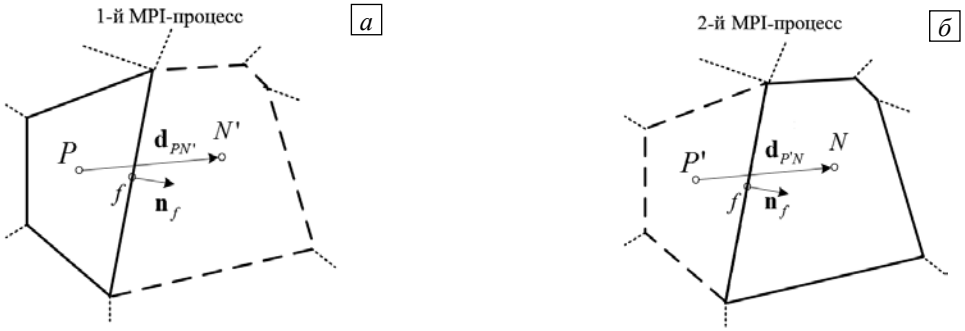

Рис. 2. Фиктивные ячейки (выделены пунктиром) на соседних процессах: на 1-м (*а*); на 2-м (*б*)

При условии $\mathbf{d}_{PN'} = \mathbf{d}_{P'N}$, согласно равенству ячеек P и P', N и N', получаем: $(\varphi_f)_{1\text{процесс}} = (\varphi_f)_{2\text{процесс}}$. При этом расчетные алгоритмы остаются неизменными, так как грань f считается внутренней и обрабатывается таким же образом, как и все остальные внутренние грани. Другим преимуществом данного варианта является возможность расширения слоя фиктивных ячеек путем создания следующего слоя ячеек P'' и N'', причем количество дополнительных слоев может быть произвольным. Это позволяет расширить сеточной шаблон и применять схемы произвольного порядка точности, ограниченные лишь числом слоев фиктивных ячеек (необходимо отметить, что схемы выше третьего порядка точности требуют знания индексов оппозитных ячеек, и в большинстве случаев это реализуемо только в структурированных сеточных моделях). Именно этот вариант перехода между MPI-процессами используется при параллельном счете задач, приведенных в данной статье, и он полностью адаптирован к произвольным неструктурированным сеткам.

Рассмотрим процесс формирования одного слоя фиктивных ячеек P' и N' более детально и условно выделим три основных этапа. На первом этапе выполним декомпозицию расчетной области на домены (один домен на область для любого MPI-процесса), при этом каждая расчетная ячейка отображается на конкретный MPI-процесс (см. Рис. 3*а*).

На данном этапе используются общепринятые подходы, и в случае декомпозиции неструктурированной многогранной сетки считается ее отображение тем или иным образом на неориентированный граф. Задача оптимального разбиения произвольного графа на части является NP-полной [27], то есть задачей, решение которой требует времени, полиномиально зависящего от размера входных данных, что приводит к экспоненциальному росту объема вычислений с увеличением размерности графа и ограничивает практическую полезность точных методов, основанных на динамическом программировании [28]. Уже при числе вершин порядка сотни точное решение задачи разбиения графа в общем случае за обозримое время не находится [27]. Возникает необходимость в привлечении различных эвристических алгоритмов, широко практикуемых при разбиении графов. В этом случае наиболее привлекательны, с точки зрения правовых ограничений, алгоритмы из программного комплекса Chaco [29, 30], свободно распространяемого по лицензии GNU LGPL, что позволяет прибегать к нему в любых коммерческих программах без

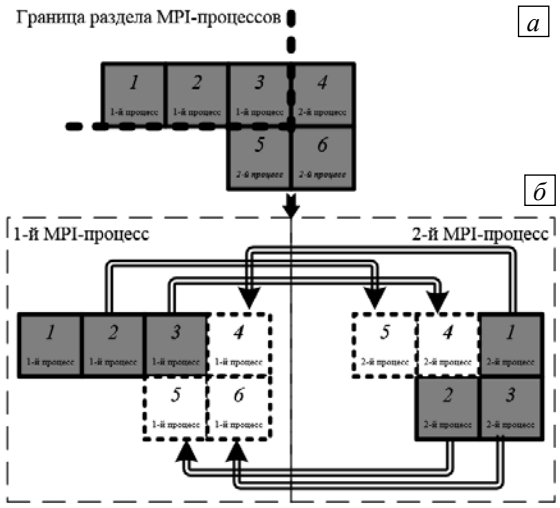

Рис. 3. Расчетная область, состоящая из шести контрольных объемов (*а*); формирование фиктивных ячеек при декомпозиции области с помощью программного комплекса Chaco (*б*); стрелками обозначены направления пересылки между процессами значений расчетных величин

раскрытия исходных текстов. Именно данный программный комплекс служил для декомпозиции в приведенных далее задачах. Как было сказано выше, результат данного этапа — сформированный список ячеек, принадлежащих тому или иному MPI-процессу (см. Рис. 3*а*).

Наиболее важный, второй этап, включает создание слоя фиктивных ячеек. Алгоритм при этом реализуется таким образом, что количество слоев может задаваться произвольно, и второй слой фиктивных ячеек образуется на основе первого, третий слой — на основе второго и так далее. В списке фиктивных ячеек учитывается узловое соседство (соседство через узлы сеточной модели), то есть статус фиктивной ячейки определяется по наличию общих с расчетной ячейкой узлов.

Сеточная модель строится следующим образом. Сначала сохраняются координаты узлов, каждому узлу присваивается уникальный порядковый номер. На основании номеров узлов формируются грани путем составления списка из номеров узлов по часовой стрелке (важно: список не произвольный), то есть грань — это список номеров узлов. Каждой грани также дается уникальный номер. И в конце на основании списка граней строятся ячейки. Такой способ, без ограничений на параллельность, открывает путь к геометрическим схемам восстановления величин на гранях, например, не по значениям в центрах ячеек, а по значениям в узлах (при этом значение в узле вычисляется по всем узлам из окружающих контрольных объемов, а значение на грани — по формирующим ее узлам). В случае фиктивных ячеек, проходящих через смежные грани, применение подобных схем в параллельном режиме становится невозможным. Хотя узловое соседство и увеличивает объем информации при межпроцессных обменах, но гарантирует универсальность при параллельном использовании произвольных геометрических схем.

Продемонстрируем декомпозицию расчетной области, состоящей из шести контрольных объемов–ячеек (Рис. 3). На рисунке изображен случай, когда на разных процессах количество фиктивных и счетных ячеек не совпадает. Часть рисунка 3*а* включает границу раздела MPI-процессов. Видно, что на 1-м процессе, согласно межпроцессной линии, в декомпозиции участвуют ячейки *2* и *3*. Соответственно слой фиктивных ячеек для 1-го процесса содержит все ячейки со смежными узлами, то есть ячейки с глобальными номерами *4*, *5*, *6*, со стороны 2-го процесса — фиктивные ячейки *2* и *3*. Таким образом, объем пересылаемой информации между процессами не всегда равномерный, но, как будет показано ниже, это практически не влияет на эффективность параллельной реализации.

На заключительном этапе фиктивные ячейки сортируются по номерам процессов–«соседей» и добавляются в конец списка основных счетных ячеек. Результаты сохраняются в соответствующих файлах декомпозиции для каждого процесса и содержат информацию о сеточном домене (списки узлов, граней, счетных и фиктивных ячеек) и формальные правила приема и передачи межпроцессных сообщений.

На основании правил третьего этапа, при необходимости, осуществляются межпроцессные обмены. Для обмена расчетными величинами на этапах итерационного цикла вызываются асинхронные MPI-функции приема и передачи сообщений (авторам настоящей статьи, благодаря личному опыту, известно, что использование синхронных функций обмена приводит к общему замедлению счета на 5÷10%), которые присутствуют в процедуре, производящей обмен массивами данных со всеми соседними MPI-процессами одновременно. Если номер текущего процесса больше, чем номер соседнего процесса, то сначала происходит посылка данных, а потом прием; при обратном соотношении номеров процессов направление обмена меняется на противоположное. Процедура передачи, содержащая целочисленные, вещественные и векторные данные, ассоциированные с ячеечными массивами, рассылает

информацию из граничных расчетных ячеек каждого процесса в фиктивные ячейки всех соседних MPI-процессов (Рис. 3). Механизм обмена ячеечными массивами одинаков вне зависимости от числа процессов.

Необходимо отметить, что все этапы декомпозиции реализованы в последовательном режиме и делаются один раз для каждой задачи с передачей необходимой информации в распределенные по процессам файлы. Это позволяет значительно сократить время повторного счета задач, для которых информация о декомпозиции берется из ранее сохраненных файлов.

С учетом вышеизложенного параллельная версия процедуры SIMPLE отличается от исходной последовательной версии только включением межпроцессных обменов и содержит аналогичные шаги:

1. Вычисление градиентов трех компонент скоростей и давления с межпроцессными обменами векторными градиентами величин.
2. Решение уравнения сохранения количества движения счетных и фиктивных ячеек для определения предварительного поля скорости, распределенного по процессам. Межпроцессные обмены массивами значений скоростей выполняются на каждой итерации линейного решателя до достижения сходимости.
3. Нахождение массовых потоков на гранях контрольного объема. Данные для установления потока на гранях, разделяющих счетную и фиктивную ячейки, берутся у фиктивных ячеек, информация для которых появилась после межпроцессных обменов на шаге 1.
4. Расчет поля приращения давления на основании уравнения (6) для счетных и фиктивных ячеек. Межпроцессные обмены производятся аналогично шагу 2 на каждой итерации линейного решателя.
5. Отыскание градиента приращения давления и корректировка поля скорости по уравнению (5), перерасчет массового потока на гранях, удовлетворяющего исходному уравнению неразрывности. Осуществление межпроцессных обменов данными о градиенте приращения давления и подправленном поле скорости.
6. Проверка условия выхода из счета (например, по величине невязки расчетных полей) и при необходимости возврат к шагу 1. Проведение коллективных межпроцессных операций суммирования.

При наличии дополнительных уравнений число межпроцессных обменов ячеечными массивами соответствующих полей увеличивается, они производятся на каждом итерационном шаге алгоритма SIMPLE, а также при решении соответствующих СЛАУ в количестве, равном количеству внутренних итераций, необходимых для выполнения условия сходимости.

Представленный параллельный алгоритм демонстрирует, что обмены между различными процессами происходят практически на каждом шаге алгоритма, но основное количество приходится на этап решения СЛАУ. Этот факт подтверждается простыми вычислениями. Так, общее количество межпроцессных обменов только для алгоритма SIMPLE равно восьми (четыре — для обмена градиентами скорости и давления, один — для градиента поправки давления и три — для подправленной скорости). Для решателя СЛАУ эта цифра намного больше, и если, например, выполнить три итерации на многосеточном решателе с построением десяти грубых уровней и сделать по три итерации на каждом уровне, то получится сто восемьдесят межпроцессных обменов при использовании V-цикла. И хотя эта оценка достаточно грубая, но она показывает, что важным фактором реализации параллельного алгоритма SIMPLE является эффективность, с которой решается СЛАУ, и именно этот — ключевой — этап требует минимизации количества межпроцессных обменов внутри решателя, что в итоге повысит производительность алгоритма в целом.

4. Организация решения линейных уравнений

В зависимости от задачи и метода решения СЛАУ количество внутренних линейных итераций может варьироваться от одной до десятков тысяч. В алгоритме SIMPLE матрица СЛАУ для давления часто не имеет строгого диагонального преобладания, и ее число обусловленности может достигать величины $10^7 \div 10^9$, что затрудняет решение СЛАУ итерационными методами [18, 31] в подпространствах Крылова и требует затрат более 90% времени на расчетном шаге. В программном комплексе ЛОГОС для решения наиболее сложной СЛАУ для давления используется алгебраический многосеточный метод (AMG), детали реализации которого подробно изложены в [18, 31, 32], а для решения СЛАУ для скоростей и турбулентных параметров — симметричный решатель Гаусса–Зейделя.

Основная идея многосеточного метода заключается в иерархическом построении и сохранении последовательности вложенных грубых матриц СЛАУ и операторов произвольного перехода от одной матрицы СЛАУ к другой. В зависимости от параметров огрубления количество этих грубых матриц (уровней) может быть произвольным. Процесс решения начинается с исходной СЛАУ (с 0-го уровня) и последующей интерполяции найденного решения и невязки на более грубые уровни. При достижении самого грубого уровня процесс интерполяции решения и невязки начинается в обратном направлении. Такая итерационная процедура позволяет значительно сократить количество внутренних итераций до момента получения необходимой точности решения.

Рассмотрим основные шаги алгебраического метода решения СЛАУ на классической задаче:

$$A_h x_h = b_h ,$$

где A_h — начальная исходная матрица размером $n \times n$; x_h и b_h — вектор неизвестных и правая часть системы уравнений, соответственно, размерами n; нижний индекс h указывает на принадлежность уравнений к подробной сетке.

Оператор интерполяции P с грубой сетки H на подробную сетку h дает возможность представить матрицу A_H на грубой сетке в виде:

$$A_H = RA_hP,$$

где $R = P^{\mathrm{T}}$. Шаг коррекции решения составляет:

$$x_h^{new} = x_h^{old} + Pe_H.$$

Коррекция решения e_H является точным решением уравнения

$$A_H e_H = r_H,$$

в котором $r_H = Rr_h$ — невязка на грубом уровне и $r_h = b_h - A_h x_h^{old}$ — невязка на самом подробном уровне.

Таким образом, одна итерация многосеточного метода со схемой коррекции решения включает последовательность шагов:

1. Выполнение n итераций предварительного сглаживания решения на сетке h при помощи метода Гаусса–Зейделя (см. описание ниже).
2. Вычисление невязки $r_h = b_h - A_h x_h^{old}$ на текущем уровне.
3. Нахождение приближенного решения $A_H e_H = r_H$ на грубой сетке. Для этого рекурсивно делается γ циклов многосеточного метода.
4. Интерполяция подправленного e_H на подробную сетку и коррекция решения на подробной сетке: $x_h^{new} = x_h^{old} + Pe_H$.
5. Осуществление n итераций заключительного сглаживания решения на подробной сетке для подавления ошибки интерполяции.

В зависимости от числа γ рекурсивных вызовов метода на каждом сеточном уровне выделяются различные типы циклов. При $\gamma = 1$ имеет место V-цикл, а при $\gamma = 2$ — W-цикл. Если на каждом уровне рекурсивно вызывать сначала один W-цикл, а затем V-цикл, получится F-цикл. Использование того или иного типа цикла (V, W и F-цикл), как показано в работе [33], влияет на скорость сходимости и устойчивость решения.

Важным фактором эффективной реализации многосеточного решателя является сглаживатель. В данной работе роль основного сглаживателя выполняет симметричный метод Гаусса–Зейделя [31]:

$$\left.\begin{aligned} x_i^{new} &= \frac{\widetilde{b}_i - \sum_{j=i+1}^{N} a_{ij} x_i^{old}}{a_{ii}}, \\ \widetilde{b}_j &= \widetilde{b}_j - a_{ji} x_i^{old} \quad (j \in i-1,\ldots,N) \end{aligned}\right\} \quad i \in 1,\ldots,N.$$

Итерационная процедура многосеточного решателя совместно со сглаживателем Гаусса–Зейделя позволяют значительно сокращать время решения СЛАУ для давления в итерационном алгоритме SIMPLE, что продемонстрировано в [20].

Основная счетная нагрузка в последовательном и параллельном вариантах многосеточного решателя ложится на создание грубых уровней и на итерационный вызов вычислительной процедуры сглаживателя. Если параллельное построение грубых уровней обсуждено в [20], то осуществление параллельных итераций сглаживателем освещено достаточно слабо. При введении фиктивных ячеек существенным плюсом является возможность избавиться от дополнительных обменов при матрично-векторных операциях. Покажем это на примере.

Рассмотрим более подробно распределенное хранение и параллельное умножение матрицы на вектор. Эффективная параллельная реализация на представленной структуре данных предполагает построение матрицы каждым MPI-процессом как для счетных, так и фиктивных ячеек. На рисунке 4 показаны портреты матрицы СЛАУ, созданные для уравнения неразрывности на восьми ячейках расчетной области при двух процессных вариантах. Межпроцессная граница разделяет при этом ячейки под номерами *2* и *3*, *6* и *7*. Найденные при линеаризации уравнений внедиагональные коэффициенты матрицы, отвечающие

за связь между ячейками, выделены в виде одинаковых фигур. Введение фиктивных ячеек гарантирует равенство коэффициентов (см. (7)), и поэтому операция умножения локальных матриц на локальные части распределенного вектора приводит к глобальному вектору, в точности такому же, как на одном процессе.

Предложенный подход открывает путь, например, в многосеточном решателе СЛАУ, имеющем в основе сглаживатель Гаусса–Зейделя, к использованию только одного межпроцессного обмена для каждой внутренней итерации, то есть при вычислении значений x^{n+1} и b в прямом и обратном ходах, а также при определении невязки $r = b - Ax$ межпроцессные обмены отсутствуют. В итоге сглаживателю методом Гаусса–Зейделя потребуется только один межпроцессный обмен информацией о фиктивном фрагменте вектора неизвестных x^{n+1} в конце единичной внутренней итерации.

Также можно утверждать, что данный подход к параллельному решению СЛАУ позволяет избавиться от дополнительного межпроцессного обмена данными о фиктивном фрагменте вектора неизвестных при «получении» решения из решателя СЛАУ, так как этот обмен уже произошел непосредственно в линейном решателе.

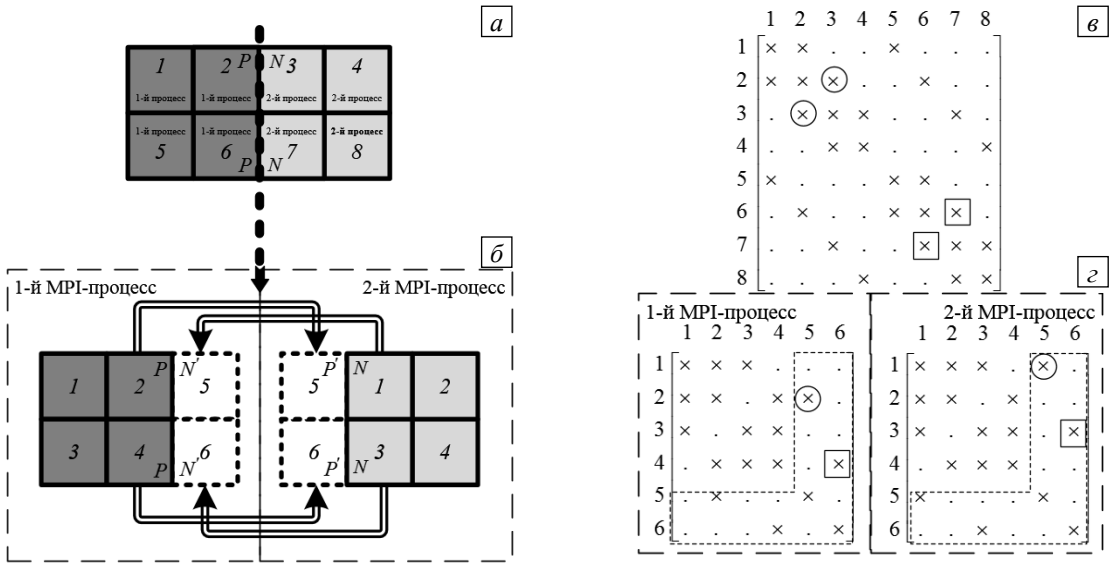

Рис. 4. Пример декомпозиции сеточной модели (*а*), (*б*) и портреты матриц СЛАУ в однопроцессорном (*в*) и двухпроцессорном (*г*) вариантах; элементы фиктивной матрицы выделены пунктиром, символы означают: «×» – наличие связи между ячейками (см. номера строк и столбцов), символ «•» – отсутствие связи; элементы матрицы на 1-м и 2-м процессах, взятые в кружки и квадратики, равны между собой соответственно

Следующий шаг на пути к эффективной реализации решателя СЛАУ — это настройка решателя AMG таким образом, чтобы количество внутренних итераций многосеточного решателя СЛАУ стало минимальным. Чем меньше внутренних итераций в решателе СЛАУ, тем меньше межпроцессных обменов, увеличивающих время расчета. Основными настройками, влияющими на количество внутренних итераций многосеточного решателя, являются:

1. Тип используемого цикла (V, W или F-цикл).
2. Количество итераций сглаживателя на каждом уровне AMG.
3. Количество ячеек для огрубления (две, четыре, восемь).

Для обоснования выбора оптимальных настроек многосеточного решателя СЛАУ, обеспечивающих минимальное количество итераций, приведем результаты численных экспериментов в следующих задачах: течение в замкнутой каверне с движущейся верхней крышкой (Задача 1) [34], турбулентное обтекание плоской пластины (Задача 2) [35] и течение в канале с обратным уступом (Задача 3) [36]. Все задачи согласно исходной постановке — двумерные, но решались в трехмерной постановке. Количество ячеек задачах, соответственно, равнялось: $1{,}0 \cdot 10^3$; $3{,}4 \cdot 10^3$; $8{,}6 \cdot 10^3$. В качестве сглаживателя применялся симметричный алгоритм Гаусса–Зейделя как наименее затратный по числу машинных операций на одной итерации [37, 38]. Рекурсивное огрубление производилось до тех пор, пока на самом грубом уровне оставалось максимум 5 счетных ячеек. Задачи решались в последовательном режиме (на одном процессе), и цель данного численного эксперимента состояла только в определении оптимальных настроек многосеточного решателя. По этой причине все задачи содержали «небольшое» количество ячеек и могли достаточно быстро решаться на современном персональном компьютере средней производительности в течение нескольких минут. Все полученные результаты по каждой изменяемой настройке сведены далее в три таблицы.

Таблица 1 содержит полное количество внутренних итераций для каждого из V, W и F циклов, необходимых сглаживателю в многосеточном решателе до полной сходимости задачи, реализуемой согласно алгоритму SIMPLE. Также в таблице приведено общее время счета в стационарной постановке до сходимости по массе порядка 10^6, вычисляемой по формуле:

$$res_m = \sum_{i=1}^{N} \left| \sum_{f=nb(P)} \rho_f S_f \left(\mathbf{u}_f \cdot \mathbf{n}_f \right) \right|,$$

здесь суммирование осуществляется по всем N счетным ячейкам модели.

Таблица 1. Результаты расчетов с циклами V, W и F

Задача	Полное количество итераций			Время счета, с		
	V-цикл	W-цикл	F-цикл	V-цикл	W-цикл	F-цикл
Задача 1	15633	492093	45090	8,9	27,4	12,3
Задача 2	48384	1034775	107730	17,4	48,5	20,0
Задача 3	61416	1354076	198378	28,3	89,2	38,8

Из таблицы однозначно видно, что применение V-цикла в многосеточном решателе СЛАУ существенно сокращает общее количество итераций и время решения задач. По сравнению с V-циклом количество итераций для F-цикла выше в 2÷3 раза, а для W-цикла — в 20÷30 раз. Общее время счета также минимально при использовании V-цикла.

Далее будем рассматривать только V-цикл. Общее количество V-циклов в решении задачи зависит от количества итераций сглаживателя на каждом уровне многосеточного метода. Чем больше их число, тем меньше V-циклов потребуется для полного решения. Однако с увеличением количества итераций сглаживателя многосеточный решатель «вырождается» в решатель Гаусса–Зейделя, что приводит к естественному замедлению счета. В случае высокопараллельных вычислений замедление становится более существенным, так как имеет место увеличение количества межпроцессных обменов. Таким образом, необходимо знать оптимальное количество шагов сглаживателя на каждом грубом уровне для получения минимального времени счета. Таблица 2 содержит данные по влиянию изменения количества итераций сглаживателя на каждом уровне на время решения представленных задач.

Таблица 2. Зависимость числа V-циклов от количества итераций сглаживателя на иерархических уровнях многосеточного метода

Задача	Количество итераций сглаживателя						
	1	2	3	4	5	6	8
	Время счета, с / полное число V-циклов						
Задача 1	10,7 / 1553	8,6 / 769	8,8 / 579	9,2 / 501	9,9 / 472	11,2 / 470	13,1 / 441
Задача 2	28,9 / 9773	25,6 / 5104	22,2 / 3270	22,3 / 2431	22,6 / 2056	23,3 / 1807	27,5 / 1611
Задача 3	27,1 / 10120	24,9 / 5011	24,4 / 3448	24,7 / 2796	25,4 / 2477	26,8 / 2225	30,1 / 1956

Анализ содержимого данной таблицы не дает однозначного ответа на вопрос о принципе выбора того или иного количества итераций сглаживателя на каждом уровне. Применение двух или трех итераций обеспечивает минимальное физическое время счета хотя бы одной задачи. Использование одной и более четырех итераций сглаживателя приводит к замедлению счета, и при их числе, равном шести, в первой задаче замедление достигает 30%. Чуть меньшее замедление, порядка 20%, наблюдается в Задачах 2 и 3, но уже на восьми итерациях сглаживателя. Понятно, что с ростом числа итераций сглаживателя время решения будет увеличиваться, и задание количества итераций сглаживателя в общем случае индивидуально для каждой задачи. Например, для Задачи 2 можно взять как три, так и четыре или пять итераций сглаживателя.

Дальнейшее ускорение вычислений многосеточным методом связано с выбором степени огрубления (оптимального количества ячеек, образующих грубую ячейку), влияющей на число грубых уровней и суммарное число грубых ячеек. Чем выше данный параметр, тем меньше уровней и грубых ячеек генерируется, и наоборот. Задача состоит в определении оптимальной степени огрубления, так как чем она выше, тем меньше грубых ячеек и их уровней нужно создавать. Следовательно, требуется меньшее время на построение грубых уровней и выполнение одной итерации AMG (то есть одного V-цикла), но тем больше итераций сглаживателя и итераций AMG необходимо для решения задачи, так как при уменьшении числа грубых уровней многосеточный решатель постепенно вырождается в обычный решатель Гаусса–Зейделя. Таблица 3 содержит данные о влиянии количества ячеек для огрубления на общее число итераций сглаживателя и физическое время счета задачи.

Таблица 3. Результаты расчетов при различном количестве ячеек для огрубления

Задача	Количество ячеек для огрубления			Количество ячеек для огрубления		
	2	4	8	2	4	8
	Число итераций сглаживателя			Время, с		
Задача 1	13842	13296	14430	9,2	8,0	8,6
Задача 2	41472	41832	48656	24,7	21,3	24,8
Задача 3	51052	50110	51370	30,7	25,3	25,7

Необходимо отметить, что при увеличении количества ячеек для огрубления общее количество уровней уменьшается, и, как показано в таблице 3, количество итераций и время счета задачи начинают расти при использовании 8 ячеек. В данных тестовых случаях оптимальным вариантом будет огрубление по четырем ячейкам для генерации следующих грубых уровней.

Общий анализ полученных результатов свидетельствует, что для оптимальной работы многосеточного решателя в настройках необходимо: задавать V-цикл; три итерации сглаживателя на каждом уровне; огрублять по четырем ячейкам. Опыт решения производственных задач позволяет заключить, что данные настройки не универсальны (например, при решении сжимаемых задач целесообразней прибегать к F-циклу по причине его большей устойчивости, что по умолчанию и делается в пакете программ ЛОГОС), и для некоторых задач более оптимальными могут быть другие типы циклов и иное количество ячеек для огрубления. Стоит отметить, что наиболее устойчивое решение СЛАУ обеспечивается при F-цикле, трех итерациях сглаживателя и двух ячейках для огрубления, при этом, как показано выше, время решения СЛАУ может увеличиться до 3 раз. Данные настройки совместно с решением системы линейных уравнений до относительной точности порядка 0,1 (относительная точность — отношение текущей невязки к начальной) и ограничением максимального числа итераций до 30 (в случае недостижения заданной точности выход из решателя СЛАУ осуществляется после 30-й итерации) [18, 33] позволяют получать минимальное количество внутренних циклов и, соответственно, минимальное время решения большинства задач гидродинамики с использованием алгоритма SIMPLE. Анализ применения различных методов ускорения газодинамических расчетов детально обсуждается в [33].

5. Способы оценки эффективности

Эффективность работы расчетного алгоритма SIMPLE измерялась двумя различными способами — прямо и косвенно. Прямая (реальная) оценка эффективности основывается на сравнении времени расчета задачи на одном процессоре (T_1) со временем расчета этой же задачи на нескольких процессорах. Так, ускорение при параллельной реализации на p процессорах составляет:

$$S_P = T_1 / T_p \ ,$$

Для оценки масштабируемости параллельного алгоритма используется понятие расчетной эффективности:

$$E_P = \left(S_p / p \right) \cdot 100\% \ .$$

Понятно, что в реальных расчетах из-за наличия межпроцессных обменов эффективность $E_P = 100\%$ недостижима.

В настоящее время существует большое разнообразие программ косвенной (программной) оценки эффективности параллельной реализации алгоритмов (см., например, [39]) на нескольких процессорах без проведения экспериментов на одном процессоре. Здесь с этой целью прибегали к программному средству STK (Statistics Tool Kit) [39], позволяющему оценить эффективность параллельных алгоритмов, которые используют MPI-интерфейс передачи сообщений [40] и OpenMP-интерфейс [19] многопоточного распараллеливания на вычислительных узлах с общей памятью. Программа STK делает возможным проведение мониторинга и анализа эффективности использования ресурсов многопроцессорных ЭВМ. Помимо этого в STK имеются средства для определения причин неэффективной работы как программы в целом, так и отдельных ее фрагментов.

В STK эффективность выполнения задачи подсчитывается как отношение времени вычислений ко всему времени счета. Для каждого MPI-процесса параллельной задачи измеряется время выполнения вызовов функций MPI и ввода–вывода ($T_{\text{обмена}}$), а также общее время ($T_{\text{общее}}$) работы программы. Исходя

из этих данных можно находить показатель эффективности функционирования параллельной программы на отдельном MPI-процессе по формуле:

$$E_i^{STK} = \frac{T_{\text{а}}}{T_{\text{а}} + T_{\text{обмена}}} \cdot 100\% ,$$

где $T_{\text{а}} = T_{\text{общее}} - T_{\text{обмена}}$ — время чисто арифметической работы на процессе с i-м номером. Для оценки эффективности выполнения параллельной программы определяется средняя эффективность выполнения всех MPI-процессов:

$$E_{avr}^{STK} = \frac{\sum_{i=1}^{N} T_{\text{а}}}{\sum_{i=1}^{N} \left(T_{\text{а}} + T_{\text{обмена}} \right)} \cdot 100\% ,$$

где N — количество процессов, E_{avr}^{STK} — показатель средней программной эффективности выполнения параллельной задачи.

В эффективном параллельном алгоритме при условии равномерной декомпозиции (количество ячеек на каждом MPI-процессе одинаково) все MPI-процессы должны затрачивать одно и то же время на решение своего фрагмента задачи. Однако в реальных условиях это труднодостижимо (из-за наличия разномасштабных ячеек, ввода–вывода, сопряженного теплообмена и другого). Поэтому для каждого из MPI-процессов устанавливается свой показатель эффективности E_i^{STK}, и среди них выделяются минимальный — $E_{\min}^{STK}$, максимальный — $E_{\max}^{STK}$, и средний — E_{ave}^{STK}, показатели программной эффективности. Вместе они дают представление о дисбалансе вычислений параллельной задачи. Близость E_{avr}^{STK} к $E_{\max}^{STK}$ свидетельствует о слабом дисбалансе (небольшое число MPI-процессов участвует меньше, чем остальные), близость E_{avr}^{STK} к $E_{\min}^{STK}$ при $E_{avr}^{STK} \ll E_{\max}^{STK}$ говорит о сильном дисбалансе (на небольшое число MPI-процессов приходится существенно большая часть вычислений, чем на остальные).

6. Численные эксперименты

Проведенные численные эксперименты направлялись в первую очередь на определение эффективности параллельной реализации алгоритма SIMPLE. Поэтому размерность задач выбиралась достаточно большой, чтобы имелась возможность распараллелить данные задачи на сотни процессов. Настройки многосеточного решателя выбирались исходя из численных экспериментов, приведенных выше, справедливых также и при работе в параллельном режиме. Декомпозиция представленных тестовых задач включала разное число процессов, причем количество процессов, участвующих в счете, увеличивалось до «насыщения», то есть до момента, когда время счета становилось большим по сравнению с предыдущим значением. Это наглядно показывало границу перехода к неэффективным вычислениям, и нецелесообразному дальнейшему увеличению числе процессов.

Для того чтобы оценить эффективность на разном числе процессов, были выбраны области из $6{,}70 \cdot 10^5$, $4{,}39 \cdot 10^6$, $1{,}439 \cdot 10^7$ ячеек. Выбор их количества обуславливался физикой течения. Задачи 1 и 3 — это внутренние течения, которые широко распространены практически во всех отраслях промышленности (автомобильной, атомной и другой). Задача 2 — внешнее обтекание тела — типичная задача автомобильной и авиационной промышленностей.

Течения во всех обсуждаемых далее численных экспериментах считались турбулентными, и для расчетов использовалась $k-\omega$ SST модель турбулентности с автоматическим определением зоны пограничного слоя [24]. Сравнение с экспериментальными или аналитическими данными для представленных задач в данной работе не приводилась, но результаты моделирования достоверны. Подробную информацию о точности представления различных течений с помощью пакета программ ЛОГОС можно найти в работах [18, 20–23].

6.1. Задача 1 — турбулентное течение жидкости в трубе круглого сечения

Рассмотрим стационарное изотермическое турбулентное течение вязкой несжимаемой жидкости в прямолинейной круглой трубе с числом Рейнольдса $\mathrm{Re} = 10^5$. Схема расчетной области представлена на рисунке 5. Для дискретизации построим блочно-структурированную сетку с общим числом ячеек 670 000.

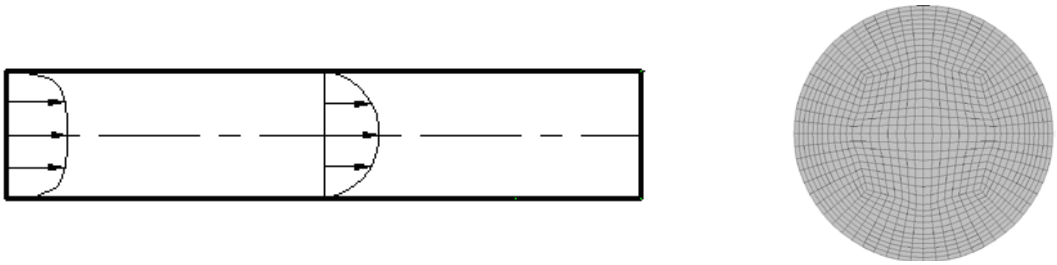

Рис. 5. Геометрия расчетной области и сеточная модель

Сравнить время решения задачи на различном количестве MPI-процессов с учетом оценок ускорения, найденным по двум методикам, можно по с помощью данных в таблице 4.

Таблица 4. Информация для оценки эффективности параллельной реализации Задачи 1

Количество MPI-процессов	Количество ячеек на один MPI-процесс	Время, с	S_P	E_P, %	$E_{\min}^{STK}$, %	$E_{\max}^{STK}$, %	E_{avr}^{STK}, %
1	670000	916,1	1,0	100,0	100,0	100,0	100,0
2	335000	503,2	1,8	91,0	97,2	97,2	97,2
4	167000	252,8	3.6	90,5	95.4	95,9	95,6
8	83750	131,5	7,0	87,1	91,8	92,2	91,9
16	41875	68.6	13,4	83,5	87,9	88,4	88,1
32	20938	44,5	20,6	64,3	67,8	68,1	67,9
64	10469	44,7	20,5	32,0	33,8	33,9	33,8
128	5235	49,7	18,4	14,4	19,9	21,4	20,7

Идеальное ускорение недостижимо в реальных расчетах ввиду выполнения необходимых межпроцессных обменов, нелинейно увеличивающихся с ростом числа расчетных MPI-процессов, роста числа фиктивных ячеек, расчет в которых отчасти дублируется (например, для подобласти из 10000 ячеек, слой фиктивных ячеек (размер границы) составляет ~ 30% от числа всех ячеек подобласти одного MPI-процесса, что в значительной степени предопределяет разницу между реальной эффективностью и эффективностью по STK в зависимости от уменьшения размера подобластей, так как по STK вычисления в фиктивных и нефиктивных ячейках неотличимы). В результате это приводит к замедлению расчета задачи в целом. Найти баланс между ускорением и эффективностью использования вычислительных ресурсов —задача трудоемкая и зачастую требующая индивидуального подхода к каждой конкретной ситуации.

В Задаче 1 анализ результатов показывает, что, например, реальная эффективность E_P равняется 64,3% и обеспечивается при декомпозиции на 32 MPI-процесса, при этом средняя программная эффективность E_{avr}^{STK} составляет 67,9%. Дальнейшее увеличение числа процессоров приводит к заметному снижению эффективности. Так, при декомпозиции на 64 процесса реальная эффективность E_P уже 32%, а программная $E_{avr}^{STK} = 33{,}8\%$. Отличия связаны с обработкой фиктивных ячеек, увеличивающей вычислительную нагрузку на процессоры. При отсутствии фиктивных ячеек показатель программной эффективности был бы ниже. Таким образом, в данной задаче минимальное время счета достигается при количестве ячеек порядка $10^4 \div 2 \cdot 10^4$ в каждом MPI-процессе, при этом реальная эффективность на 64 MPI-процессах уменьшается в два раза по сравнению с 32 MPI-процессами, что свидетельствует о лучшем балансе вычислений при количестве ячеек, равном $2 \cdot 10^4$. В дополнение можно отметить отсутствие дисбаланса ($E_{\min}^{STK} \approx E_{\max}^{STK}$), свидетельствующее о равномерности распределения расчетной нагрузки на все MPI-процессы.

6.2. Задача 2 — турбулентное обтекание потоком воздуха препятствия «Ahmed body»

В этом численном эксперименте решим задачу турбулентного обтекания потоком воздуха препятствия вида «Ahmed body» [41]. Скорость набегающего потока на входе в расчетную область зададим равной 15 м/с, число Рейнольдса порядка — $\mathrm{Re} = 10^6$, а параметры воздуха посчитаем стандартными.

Модель расчетной области представлена на рисунке 6. Для расчетов используется неструктурированная сетка из усеченных гексаэдров. Общее число ячеек составляет порядка 5 миллионов. Постановка задачи и сеточная область полностью трехмерные.

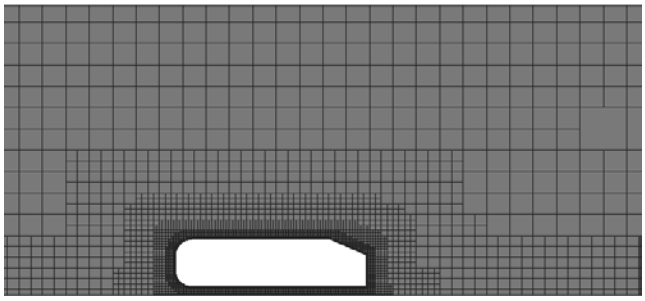

Рис. 6. Сеточная модель вокруг препятствия «Ahmed body»

Данные о времени решения задачи на различном количестве MPI-процессов с приведением оценок ускорения по двум методикам приведены в таблице 5.

Таблица 5. Информация для оценки эффективности параллельной реализации Задачи 2

Количество MPI-процессов	Количество ячеек на один MPI-процесс	Время, с	S_p	E_p, %	$E_{\min}^{STK}$, %	$E_{\max}^{STK}$, %	E_{avr}^{STK}, %
1	4392680	37063,3	1,0	100,0	100,0	100,0	100,0
2	2196340	19929,4	2,0	97,9	97,5	97,8	97,7
4	1098170	10360,6	3,8	94,0	95,8	97,5	96,7
8	549085	5636,6	7,2	90,2	93,0	95,9	94,7
16	274542	2388,0	13,8	86.2	90,1	94,5	92,3
32	173271	1271,4	27,0	84,5	89,1	91,5	90,3
64	68635	666,5	48,4	75,6	82,4	89,0	85,6
128	34317	402,5	92,1	71,9	75,3	81,7	78,0
256	17158	320,0	115,8	45,3	49,3	53.3	51,3
384	11439	288,6	128,4	33,4	38,3	39,5	38,9
512	8579	260,0	142,7	27,9	30,4	32,8	31,6
768	5720	301,6	122,9	16,0	19,9	23,8	21,9

Как видно из таблицы, минимальное время счета достигается при декомпозиции на 512 MPI-процессов, при этом реальная эффективность достаточно низкая и находится на уровне $E_P = 27{,}9\%$. Программная эффективность в данном случае составляет $E_{avr}^{STK} = 31{,}6\%$. Таким образом, в Задаче 2 оптимальное количество ячеек, рассчитываемое каждым MPI-процессом, находится в пределах от 8000 до 11000, что примерно соответствует результатам предыдущего эксперимента.

6.3. Задача 3 — течение газа в канале за обратным уступом

Исследуем стационарное изотермическое турбулентное течение газа в канале с обратным уступом, соответствующее числу Рейнольдса 21000 [24]. Для расчетов используем блочно-структурированную трехмерную сетку (Рис. 7) с общим числом ячеек 14392680. Исходную постановку задачи возьмем двумерную).

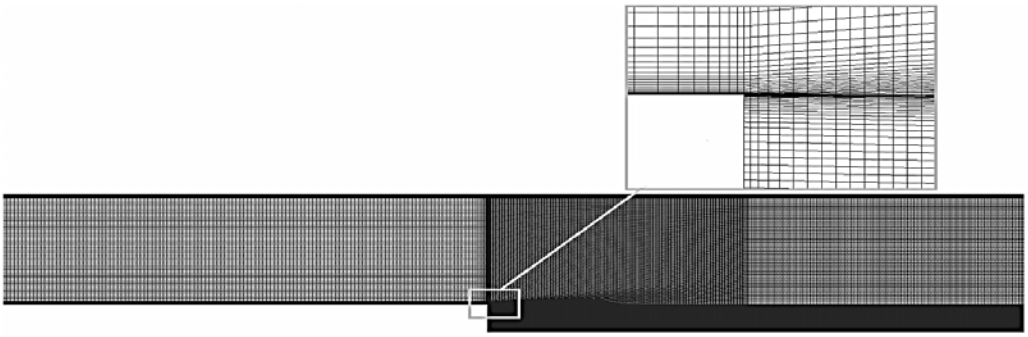

Рис. 7. Сеточная модель уступа

Для сравнения времени решения задачи на различном количестве MPI-процессов данные приведены в таблице 6. Они основаны на измерении программной эффективности при декомпозиции от 8 процессоров. Это связано с невозможностью посчитать данную задачу на одном процессе ввиду ограничения объема доступной оперативной памяти. В Задаче 3 ускорение S_p воспроизводится из E_{avr}^{STK} и является приближенной величиной с погрешностью ~ 12%, если судить по результатам из таблицы 5.

Таблица 6. Информация для оценки эффективности параллельной реализации Задачи 3

Количество процессоров	Количество ячеек на один MPI-процесс	Время, с	S_p	$E_{\min}^{STK}$, %	$E_{\max}^{STK}$, %	E_{avr}^{STK}, %
8	1799085	287523,6	7,6	92,7	98,3	95,1
16	899542	151600,5	14,9	90,8	98,0	93,0
32	449771	77656,8	28,1	85,4	92,0	87,7
64	224885	45839,4	52,6	79,7	89,7	82,2
128	112442	26081,5	104,4	78,1	85,3	81,5
256	56221	15545,0	175,1	65,6	71,6	68,4
512	28110	9580,1	284,1	53,2	58,1	55,5
1024	14055	8263,2	329,3	30,2	33,7	32,2
1280	11244	8461,7	313,7	23,2	25,7	24,5
1536	9370	9263,2	282,6	17,3	19,7	18,4

Реальную эффективность невозможно измерить, поэтому единственным средством ее оценки является программа STK. Данные, полученные при помощи такого способа, как показали расчеты в предыдущих задачах, коррелируют с расчетной эффективностью.

Как видно из таблиц, минимальному времени счета в каждом MPI-процессе отвечает ~ 14000 ячеек. При этом величины реальной и программной эффективностей находятся на уровне $E_p \approx 30\%$ и $E_{avr}^{STK} \approx 35\%$ и превышают 50% при числе ячеек более $2 \cdot 10^4$ в каждом MPI-процессе. Разница в показателях эффективности связана в первую очередь с наличием фиктивных ячеек и возрастающей нагрузкой на каждый MPI-процесс при увеличении числа процессорных ядер. Также можно с уверенностью отметить, что в каждом численном эксперименте отсутствует дисбаланс вычислений.

7. Заключение

В данной работе обсуждается параллельная реализация итерационного алгоритма SIMPLE на произвольных неструктурированных многогранных сетках. Приведено подробное описание шагов в его последовательном и параллельном вариантах. Особое внимание уделено особенностям эффективного распараллеливания многосеточного решателя СЛАУ. Предложенный способ построения локальных матриц с учетом фиктивных ячеек позволил уменьшить число межпроцессных обменов в векторно-матричных операциях. В серии численных экспериментов с вариацией настроек многосеточного решателя (а именно с применением различных типов циклов (V, W и F), изменением количества итераций сглаживателя и числа ячеек для огрубления при переходе на более грубый уровень) показано, что применение V-цикла с тремя итерациями сглаживателя и четырьмя ячейками для огрубления дают минимальное время решения задач.

На основе серии численных экспериментов установлено, что минимальное время счета задачи достигается при разбиении сетки на 10000÷20000 ячеек на процессор, независимо от размерности задачи. Также для каждой задачи продемонстрировано отсутствие дисбаланса вычислительной нагрузки процессоров.

Работа выполнена при финансовой поддержке РФФИ (проекты № 16-01-00267-а и №16-31-00080-мол_а).

Литература

1. *Ferziger J.H., Peric M.* Computational methods for fluid dynamics. – Berlin: Springer Verlag, 2002. – 423 p.
2. *Флетчер К.* Вычислительные методы в динамике жидкостей. – М.: Мир, 1991. – Т. 1. – 504 с.
3. *Patankar S.* Numerical heat transfer and fluid flow. – New York: Hemisphere Publishing Corporation, 1980. – 197 p.
4. *Moukalled F., Darwish M.A.* A unified formulation of the segregated class of algorithms for fluid flow at all speeds // Numer. Heat Tr. B-Fund. – 2000. – Vol. 37, no. 1. – P. 103-139. DOI
5. *Shterev K.S., Stefanov S.K.* Pressure based finite volume method for calculation of compressible viscous gas flows // J. Comput. Phys. – 2010. – Vol. 229, no. 2. – P. 461-480. DOI

6. *Jasak H.* Numerical solution algorithms for compressible flows: Lecture Notes. Technical report. – Croatia, Zagreb: Wikki Ltd., 2006. – 206 p.
7. *Katz A., Sankaran V.* High aspect ratio grid effects on the accuracy of Navier–Stokes solutions on unstructured meshes // Comput. Fluids. – 2012. – Vol. 65. – P. 66-79. DOI
8. *Xie B., Li S., Ikebata A., Xiao F.* A multi-moment finite volume method for incompressible Navier–Stokes equations on unstructured grids: Volume-average/point-value formulation // J. Comput. Phys. – 2014. – Vol. 277. – P. 138-162. DOI
9. *Lindholm E., Nickolls J., Oberman S., Montrym J.* NVIDIA Tesla: a unified graphics and computing architecture // IEEE Computer Society, Hot Chips19, March-April 2008. – P. 39-55.
10. *Barth M., Byckling M., Ilieva N., Saarinen S., Schliephake M., Weinberg V.* Best Practice Guide Intel Xeon Phi v1.1. – 2014. – 51 p.
11. *McDonough J.M.* Lectures in computational fluid dynamics of incompressible flow: Mathematics, algorithms and implementations. – Departments of Mechanical Engineering and Mathematics, University of Kentucky, 1991, 2003, 2007. – 155 c.
12. *Tikir M.M., Carrington L., Strohmaier E., Snavely A.* A generic algorithms approach to modeling the performance of memory-bound computations // Supercomputing, 2007. SC'07. Proc. of the ACM/IEEE Conference, November 10-16, 2007. – 12 p. DOI
13. *Emans M., Liebmann M.* Velocity-pressure coupling on GPU // SFB-Report No. 2013-003. – 2013. – 21 p.
14. *Gaburov E., Cavecchi Yu.* XeonPhi meets astrophysical fluid dynamics. www.prace-ri.eu (дата обращения 24.06.2016).
15. *Волков К.Н., Дерюгин Ю.Н., Емельянов В.Н., Карпенко А.Г., Козелков А.С., Тетерина И.В., Ялозо А.В.* Решение задач газовой динамики и теплообмена на графических процессорах // ВАНТ. Серия: Математическое моделирование физических процессов. – 2014. – № 4. – С. 22-34.
16. *Крючков И.А., Копкин С.В.* Программный комплекс моделирования методом молекулярной динамики для гибридных вычислительных систем // ВАНТ. Серия: Математическое моделирование физических процессов. – 2012. – № 1. – С. 59-65.
17. *Беляков В.А., Линев А.В., Горшков А.В., Крылов И.Б.* Моделирование релаксации массива кремниевых нанокристаллов по методу Монте-Карло с использованием графических ускорителей // Вестник ННГУ. – 2012. – № 4-1. – С. 260-267.
18. *Козелков А.С., Дерюгин Ю.Н., Лашкин С.В., Силаев Д.П., Симонов П.Г., Тятюшкина Е.С.* Реализация метода расчета вязкой несжимаемой жидкости с использованием многосеточного метода на основе алгоритма SIMPLE в пакете программ ЛОГОС // ВАНТ. Серия: Математическое моделирование физических процессов. – 2013. – № 4. – С. 44-56.
19. *Антонов А.С.* Параллельное программирование с использованием технологии OpenMP: Учебное пособие. – М.: Изд-во МГУ, 2009. – 77 с.
20. *Козелков А.С., Курулин В.В., Пучкова О.Л., Лашкин С.В.* Моделирование турбулентных течений с использованием алгебраической модели рейнольдсовых напряжений с универсальными пристеночными функциями // Вычисл. мех. сплош. сред. – 2014. – Т. 7, № 1. – С. 40-51. DOI
21. *Козелков А.С., Курулин В.В., Тятюшкина Е.С., Пучкова О.Л.* Моделирование турбулентных течений вязкой несжимаемой жидкости на неструктурированных сетках с использованием модели отсоединенных вихрей // Матем. моделирование. – 2014. – Т. 26, № 8. – С. 81-96.
22. *Козелков А.С., Шагалиев Р.М., Дмитриев С.М., Куркин А.А., Волков К.Н., Дерюгин Ю.Н., Емельянов В.Н., Пелиновский Е.Н., Легчанов М.А.* Математические модели и алгоритмы для численного моделирования задач гидродинамики и аэродинамики: Учеб. пособие. – Нижний Новгород: НГТУ им. Р.Е. Алексеева, 2014. – 164 с.
23. *Козелков А.С, Дерюгин Ю.Н., Циберева Ю.А., Корнев А.В., Денисова О.В., Стрелец Д.Ю., Куркин А.А., Курулин В.В., Шарипова И.Л., Рубцова Д.П., Легчанов М.А., Тятюшкина Е.С., Лашкин С.В., Ялозо А.В., Яцевич С.В., Тарасова Н.В., Гинниятуллин Р.Р., Сизова М.А., Крутякова О.Л.* Минимальный базис задач для валидации методов численного моделирования турбулентных течений вязкой несжимаемой жидкости // Труды НГТУ им. Р.Е. Алексеева. – 2014. – № 4 (106). – С. 21-69.
24. *Menter F.R.* Two-equation eddy-viscosity turbulence models for engineering applications // AIAA J. – 1994. – Vol. 32, no. 8. – P. 1598-1605. DOI
25. *Chorin A.J.* A numerical method for solving incompressible viscous flow problems // J. Comput. Phys. – 1967. – Vol. 2, no. 1. – P. 12-26. DOI
26. http://openfoam.org/ (дата обращения 29.06.2016).
27. *Якобовский М.В.* Обработка сеточных данных на распределенных вычислительных системах // ВАНТ. Серия: Математическое моделирование физических процессов. – 2004. – № 2. – С. 40-53.
28. *Евстигнеев В.А.* Применение теории графов в программировании / Под ред. А.П. Ершова. – М.: Наука, 1985. – 352 с.
29. *Hendrickson B., Leland R.* The Chaco user's guide: Version 2.0. – Technical Report, SAND95-2344, Sandia National Laboratories, Albuquerque, NM, July 1995. – 44 p.
30. *Leland R., Hendrickson B.* An empirical study of static load balancing algorithms // Proc. Scalable High-Perfomance Computing Conference, IEEE, 23-25 May 1994. – P. 682-685. DOI
31. *Бахвалов Н.С., Жидков Н.П., Кобельков Г.Г.* Численные методы. – М.: БИНОМ. Лаборатория знаний, 2003. – 630 с.
32. *Волков К.Н., Дерюгин Ю.Н., Емельянов В.Н., Козелков А.С., Тетерина И.В.* Алгебраический многосеточный метод в задачах вычислительной физики // Вычислительные методы и программирование. – 2014. – Т. 15, № 2. – С. 183-200.
33. *Волков К.Н., Дерюгин Ю.Н., Емельянов В.Н., Карпенко А.Г., Козелков А.С., Тетерина И.В.* Методы ускорения газодинамических расчетов на неструктурированных сетках. – М.: Физматлит, 2014. – 536 с.
34. *Ghia U., Ghia K.N., Shin C.T.* High-Re solutions for incompressible flow using the Navier–Stokes equations and a multigrid method // J. Comput. Phys. – 1982. – Vol. 48, no. 3. – P. 387-411. DOI
35. *Лойцянский Л.Г.* Механика жидкости и газа. – М.: Дрофа, 2003. – 840 с.
36. *Vogel J.C., Eaton J.K.* Combined heat transfer and fluid dynamic measurements downstream of a backward-facing step // J. Heat Transfer. – 1985. – Vol. 107, no. 4. – P. 922-929. DOI

37. *Уоткинс Д.С.* Основы матричных вычислений. – М.: БИНОМ, 2009. – 664 с.
38. *Woodward C.S., Serban R.* SUNDIALS: Suite of Nonlinear and Differential/Algebraic Equation Solvers. – Livermore: UCRL-PRES-213978.
39. *Новаев Д.А., Бартенев Ю.Г., Липов Д.И., Колпаков С.И., Киселев А.Б., Серова Т.Н., Худякова Л.В.* Программные средства STK для исследования эффективности выполнения параллельных приложений // ВАНТ. Серия: Математическое моделирование физических процессов. – 2011. – № 4. – С. 72-81.
40. *Snir M., Otto S., Huss-Lederman S., Walker D., Dongarra J.* MPI: The complete reference. – MIT Press, 1996.
41. *Ahmed S.R., Ramm G., Faltin G.* Some salient features of the time-averaged ground vehicle wake. – SAE Technical Paper 840300, 1984. – 34 p. DOI
42. *Козелков А.С., Крутякова О.Л., Куркин А.А., Курулин В.В., Тятюшкина Е.С.* Зонный RANS–LES подход на основе алгебраической модели рейнольдсовых напряжений // МЖГ. – 2015. – № 5. – С. 24-33. (English version DOI).
43. *Козелков А.С., Курулин В.В.* Численная схема для моделирования турбулентных течений несжимаемой жидкости с использованием вихреразрешающих подходов // ЖВММФ. – 2015. – Т. 55, № 7. – С. 1255-1266. (English version DOI).
44. *Kozelkov A., Kurulin V., Emelyanov V., Tyatyushkina E., Volkov K.* Comparison of convective flux discretization schemes in detached-eddy simulation of turbulent flows on unstructured meshes // Journal of Scientific Computing. – 2016. – Vol. 67, no. 1. – P. 176-191. DOI
45. *Козелков А.С., Куркин А.А., Курулин В.В., Пелиновский Е.Н., Тятюшкина Е.С.* Моделирование возмущений в озере Чебаркуль при падении метеорита в 2013 году // МЖГ. – 2015. – № 6. – С. 134-143.
46. *Козелков А.С., Куркин А.А., Пелиновский Е.Н.* Влияние угла входа тела в воду на высоты генерируемых волн // МЖГ. – 2016. – № 2. – С. 166-176.

References

1. Ferziger J.H., Peric M. *Computational methods for fluid dynamics*. Berlin: Springer Verlag, 2002. 423 p.
2. Fletcher C. *Computational techniques for fluid dynamics*. Springer, Berlin, Heidelberg, 1988, Vol. 1. 401 p.
3. Patankar S. *Numerical heat transfer and fluid flow*. New York: Hemisphere Publishing Corporation, 1980. 197 p.
4. Moukalled F., Darwish M.A. A unified formulation of the segregated class of algorithms for fluid flow at all speeds. *Numer. Heat Tr. B-Fund*., 2000, vol. 37, no. 1, pp. 103-139. DOI
5. Shterev K.S., Stefanov S.K. Pressure based finite volume method for calculation of compressible viscous gas flows. *J. Comput. Phys*., 2010, vol. 229, no. 2, pp. 461-480. DOI
6. Jasak H. *Numerical solution algorithms for compressible flows:* Lecture Notes. Technical report. Croatia, Zagreb: Wikki Ltd., 2006. 206 p.
7. Katz A., Sankaran V. High aspect ratio grid effects on the accuracy of Navier–Stokes solutions on unstructured meshes. *Comput. Fluids*, 2012, vol. 65, pp. 66-79. DOI
8. Xie B., Li S., Ikebata A., Xiao F. A multi-moment finite volume method for incompressible Navier-Stokes equations on unstructured grids: Volume-average/point-value formulation. *J. Comput. Phys*., 2014, vol. 277, pp. 138-162. DOI
9. Lindholm E., Nickolls J., Oberman S., Montrym J. NVIDIA Tesla: a unified graphics and computing architecture. *IEEE Computer Society, Hot Chips19, March-April 2008. Pp. 39-55*.
10. Barth M., Byckling M., Ilieva N., Saarinen S., Schliephake M., Weinberg V. *Best Practice Guide Intel Xeon Phi v1.1*, 2014. 51 p.
11. McDonough J.M. *Lectures in computational fluid dynamics of incompressible flow: Mathematics, algorithms and implementations*. Departments of Mechanical Engineering and Mathematics, University of Kentucky, 1991, 2003, 2007. 155 p.
12. Tikir M.M., Carrington L., Strohmaier E., Snavely A. A generic algorithms approach to modeling the performance of memory-bound computations. *Supercomputing, 2007. SC'07. Proc. of the ACM/IEEE Conference, November 10-16, 2007. 12 p*. DOI
13. Emans M., Liebmann M. *Velocity-pressure coupling on GPU*. SFB-Report No. 2013-003, 2013. 21 p.
14. Gaburov E., Cavecchi Y. *XeonPhi meets astrophysical fluid dynamics*, available at: www.prace-ri.eu (accessed 24 June 2016).
15. Volkov K.N., Deryugin Yu.N., Emelyanov V.N., Karpenko A.G., Kozelkov A.S., Teterina I.V., Yalozo A.V. Gas dinamics and heat transfer simulations on graphic processing units. *VANT. Ser.: Mat. Mod. Fiz. Proc.*, 2014, vol. 4, pp. 22-34.
16. Kryuchkov I.A., Kopkin S.V. Code complex for the MD simulation on hybrid-architecture computers. *VANT. Ser.: Mat. Mod. Fiz. Proc.*, 2012, vol. 1, pp. 59-65.
17. Belyakov V.A., Linev A.V., Gorshkov A.V., Krylov I.B. Relaxation simulation of a silicon nanocrystal array by the Monte Carlo method using graphics aceelerators. *Vestnik of Lobachevsky University of Nizhni Novgorod*, 2012, no. 4-1, pp. 260-267.
18. Kozelkov A.S., Deryugin Yu.N., Lashkin S.V., Silaev D.P., Siminov P.G., Tyatyushlina E.S. Implemantation in Logos software of a computational scheme for a viscous incompressible fluid using the multigrid method based on algorithm SIMPLE. *VANT. Ser.: Mat. Mod. Phis. Proc.*, 2013, vol. 4, pp. 44-56.
19. Antonov A.S. *Parallel programming using OpenMP*: Tutorial. Moscow: Lomonosov Moscow State University, 2009. 77 p.
20. Kozelkov A.S., Kurulin V.V., Puchkova O.L., Lashkin S.V. Simulation of turbulent flows using an algebraic Reynolds stress model with universal wall functions. *Vychisl. mekh. splosh. sred – Computational Continuum Mechanics*, 2014, vol. 7, no. 1, pp. 40-51. DOI
21. Kozelkov A.S., Kurulin V.V., Tyatyushkina E.S., Puchkova O.L. Application of the detached eddy simulation model for viscous incompressible turbulent flow simulations on unstructured grids. *Matem. Mod.*, 2014, vol. 26, no. 8, pp. 81-96.
22. Kozelkov A.S. Shagaliev R.M., Dmitriev S.M., Kurkin A.A., Volkov K.N., Deryugin Yu.N., Emelyanov V.N., Pelinovskiy E.N., Legchanov M.A. Matematicheskie modeli i algoritmy dlya chislennogo modelirovaniya zadach

gidrodinamiki i aerodinamiki [Mathematical models and algorithms for numerical simulation of hydrodynamics and aerodynamics tasks]. *Nizhny Novrorod: Nizhny Novgorod State Technical University n.a. R.E. Akekseev*, 2014. 164 p.
23. Kozelkov A.S., Deryugin Yu.N., Tsibereva Yu.A., Kornev A.V., Denisova O.V., Strelets D.Yu., Kurkin A.A., Kurulin V.V., Sharipova I.L., Rubtsova D.P., Legchanov M.A., Tyatyushlina E.S., Lashkin S.V., Yalozo A.V., Yatsevich S.V., Tarasova N.V., Giniyatullin R.R., Sizova M.A., Krutyakova O.L. Minimal basis tasks for validation of methods of numerical simulation of turbulent flows of incompressible viscous fluids. *Transactions of NNSTU n.a. R.E. Akekseev*, 2014, no. 4 (106), pp. 21-69.
24. Menter F.R. Two-equation eddy-viscosity turbulence models for engineering applications. *AIAA J.*, 1994, vol. 32, no. 8, pp. 1598-1605. DOI
25. Chorin A.J. A numerical method for solving incompressible viscous flow problems. *J. Comput. Phys.*, 1967, vol. 2, no. 1, pp. 12-26. DOI
26. http://openfoam.org/ (accessed 29 June 2016).
27. Yakobovskiy M.V. Grid data processing of distributed computer systems. *VANT. Ser.: Mat. Mod. Fiz. Proc.*, 2004, vol. 2, pp. 40-53.
28. Evstigneev V.A. *Primenenie teorii grafov v programmirovanii* [Application of graph theory in programming], ed. by A.P. Ershov. Moscow: Nauka, 1985. 352 p.
29. Hendrickson B., Leland R. *The Chaco user's guide*: Version 2.0. Technical Report, SAND95-2344, Sandia National Laboratories, Albuquerque, NM, July 1995. 44 p.
30. Leland R., Hendrickson B. An empirical study of static load balancing algorithms. *Proc. Scalable High-Perfomance Computing Conference, IEEE, 23-25 May 1994. Pp. 682-685*. DOI
31. Bakhvalov N.S., Zhidkov N.P., Kobel'kov G.M. *Chislennye metody* [Numerical methods]. Moscow: BINOM, 2003. 630 p.
32. Volkov K.N., Deryugin Yu.N., Emelyanov V.N., Kozelkov A.S., Teterina I.V. An algebraic multigrid method in problems of computational physics. *Vychislitel'nye metody i programmirovanie – Numerical Methods and Programming*, 2014, no. 15, pp. 183-200.
33. Volkov K.N., Deryugin Yu.N., Emelyanov V.N., Karpenko A.G., Kozelkov A.S., Teterina I.V. Metody uskorenuya gazodinamicheskikh raschetov na nestrukturirovannykh setkakh [Methods of acceleration of gas-dynamical calculations on unstructured grids]. Moscow: Fizmatlit, 2014. 536 p.
34. Ghia U., Ghia K.N., Shin C.T. High-Re solutions for incompressible flow using the Navier–Stokes equations and a multigrid method. *J. Comput. Phys.*, 1982, vol. 48, no. 3, pp. 387-411. DOI
35. Loitsyanskii L.G. *Mechanics of liquids and gases*. Pergamon Press, 1966. 804 p.
36. Vogel J.C., Eaton J.K. Combined heat transfer and fluid dynamic measurements downstream of a backward-facing step. *J. Heat Transfer*, 1985, vol. 107, no. 4, pp. 922-929. DOI
37. Watkins D.S. *Fundamentals of matrix computations*. Wiley, 2002.
38. Woodward C.S., Serban R. *SUNDIALS: Suite of Nonlinear and Differential/Algebraic Equation Solvers*. Livermore: UCRL-PRES-213978.
39. Novaev D.A., Bartenev Yu.G., Lipov D.I., Kolpakov S.I., Kiselev A.B., Serova T.N., Hudyakova L.V. An STK software for the study of the execution efficiency for parallel applications. *VANT. Ser.: Mat. Mod. Fiz. Proc.*, 2011, vol. 4, pp. 72-81.
40. Snir M., Otto S., Huss-Lederman S., Walker D., Dongarra J. *MPI: The complete reference. MIT Press.*, 1996.
41. Ahmed S.R., Ramm G., Faltin G. *Some salient features of the time-averaged ground vehicle wake*. SAE Technical Paper 840300, 1984. 34 p. DOI
42. Kozelkov A.S., Krutyakova O.L., Kurkin A.A., Kurulin V.V., Tyatyushkina E.S. Zonal RANS-LES approach based on an algebraic reynolds stress model. *Fluid Dynamics*, 2015, vol. 50, no. 5, pp. 621-628. DOI
43. Kozelkov A.S., Kurulin V.V. Eddy-resolving numerical scheme for simulation of turbulent incompressible flows. *Comp. Math. Math. Phys.*, 2015, vol. 55, no. 7, pp. 1232-1241. DOI
44. Kozelkov A., Kurulin V., Emelyanov V., Tyatyushkina E., Volkov K. Comparison of convective flux discretization schemes in detached-eddy simulation of turbulent flows on unstructured meshes. *Journal of Scientific Computing*, 2016, vol. 67, no. 1, pp. 176-191. DOI
45. Kozelkov A.S., Kurkin A.A., Kurulin V.V., Pelinovsky E.N., Tyatyushkina E.S. Modelirovanie vozmushcheniy v ozere Chebarkul pri padenii meteorita v 2013 godu [Perturbation modeling in lake Chebarkul after meteorite impact in 2013 year]. *Fluid Dynamics*, 2015, vol. 6, pp. 134-143.
46. Kozelkov A.S., Kurkin A.A., Pelinovsky E.N. Vliyanie ugla vkhoda tela v vodu na vysoty generiruemykh voln [Influence of body angle entering the water on generated wave height]. *Fluid Dynamics*, 2016, vol. 2, pp. 166-176.

Поступила в редакцию 08.02.2016; опубликована в электронном виде 30.09.2016

Сведения об авторах

Лашкин Сергей Викторович, нач. гр., Государственная корпорация по атомной энергии «Росатом» ФГУП «РФЯЦ – ВНИИЭФ», 607188, Нижегородская обл., Саров, пр. Мира, д. 37; e-mail: lashkinsv@gmail.com

Козелков Андрей Сергеевич, кфмн, снс, нач. лаб., Государственная корпорация по атомной энергии «Росатом» ФГУП «РФЯЦ – ВНИИЭФ»; e-mail: A.S.Kozelkov@vniief.ru

Ялозо Андрей Владимирович, нс, Государственная корпорация по атомной энергии «Росатом» ФГУП «РФЯЦ – ВНИИЭФ»; A.V.Yalozo@itmf.vniief.ru

Герасимов Виталий Юрьевич, мнс, Государственная корпорация по атомной энергии «Росатом» ФГУП «РФЯЦ – ВНИИЭФ»; e-mail: V.Yu.Gerasimov@itmf.vniief.ru

Зеленский Дмитрий Константинович, нач. лаб., Государственная корпорация по атомной энергии «Росатом» ФГУП «РФЯЦ – ВНИИЭФ»; e-mail: zdk@vniief.ru