2025, **26** (4), 410–421. doi 10.26089/NumMet.v26r427

doi 10.26089/NumMet.v26r427

УДК 517.968;
519.642.3

# Алгоритм применения структурного критерия адаптации для решения задач с использованием динамически адаптивных сеток

**С. К. Григорьев**
Институт прикладной математики имени М. В. Келдыша (ИПМ РАН),
Москва, Российская Федерация
ORCID: 0009-0005-2012-6609, e-mail: sergejgri@gmail.com

**П. А. Кучугов**
Институт прикладной математики имени М. В. Келдыша (ИПМ РАН),
Москва, Российская Федерация
ORCID: 0000-0003-3240-2963, e-mail: pkuchugov@gmail.com

**Аннотация:** При решении современных задач математического моделирования широко применяется динамическая адаптация расчетной сетки. Особенности реализации численных методов при динамической адаптации требуют соблюдения определенных соотношений между линейными размерами смежных элементов, которые называются структурным критерием. Структурный критерий позволяет формировать буферный слой ячеек между областями, чей уровень измельчения отличается более чем вдвое. В работе рассматривается алгоритм формирования буферного слоя ячеек в листовой модели адаптации расчетной сетки, реализованный на распределенной вычислительной системе. Получены оценки алгоритмической сложности и эффективности алгоритма на распределенной вычислительной системе. Эксперименты показали возможность алгоритма формировать буферный слой в областях сложной формы, однако при этом его слабую масштабируемость с ростом числа процессов.

**Ключевые слова:** MPI, распределенные вычисления, octreemesh, AMR, буферный слой.

**Благодарности:** Вычисления проводились с помощью гибридного суперкомпьютера K60, установленного в Суперкомпьютерном Центре коллективного пользования ИПМ имени М. В. Келдыша РАН.

**Для цитирования:** Григорьев С.К., Кучугов П.А. Алгоритм применения структурного критерия адаптации для решения задач с использованием динамически адаптивных сеток // Вычислительные методы и программирование. 2025. **26**, № 4. 410–421. doi 10.26089/NumMet.v26r427.

2025, **26** (4), 410–421. doi 10.26089/NumMet.v26r427

# Algorithm for applying the structural adaptation criterion to solve problems using adaptive mesh refinement

**Sergey K. Grigoriev**
Keldysh Institute of Applied Mathematics of RAS, Moscow, Russia
ORCID: 0009-0005-2012-6609, e-mail: sergejgri@gmail.com
**Pavel A. Kuchugov**
Keldysh Institute of Applied Mathematics of RAS, Moscow, Russia
ORCID: 0000-0003-3240-2963, e-mail: pkuchugov@gmail.com

**Abstract:** Adaptive mesh refinement (AMR) is widely used in solving modern mathematical modeling problems. The specifics of the implementation of numerical methods for AMR require compliance with certain ratios between the linear sizes of adjacent elements, which are called the structural criterion. Structural criterion allows creation of buffer layer of cells between regions, which cell division level difference is greater than two. In this paper we discuss the algorithm for buffer layer generation for tree-based adaptive mesh refinement on distributed computational system. Estimates of the algorithmic complexity and efficiency of the algorithm on distributed computational system are obtained. Experiments have shown the possibility of algorithm for forming a buffer layer in areas of complex shape, but at the same time its poor scalability with an increasing number of processes.

**Keywords:** MPI, distributed computing, octreemesh, AMR, buffer layer.

**Acknowledgements:** The research is carried out using the K60 hybrid supercomputer installed at the Supercomputer Center for Collective Use of the Keldysh Institute of Applied Mathematics of the Russian Academy of Sciences.

**For citation:** S. K. Grigoriev, P. A. Kuchugov, "Algorithm for applying the structural adaptation criterion to solve problems using adaptive mesh refinement," Numerical Methods and Programming. **26** (4), 410–421 (2025). doi 10.26089/NumMet.v26r427.

**1. Введение.** При решении современных прикладных задач часто приходится иметь дело с многомасштабными нестационарными физическими процессами. Динамическая адаптация является одним из способов повышения точности решения при сохранении доступной вычислительной стоимости. Различают два вида адаптации: P-уточнение (P-refinement), состоящее в повышении порядка аппроксимации при сохранении подробности сетки, и H-уточнение (H-refinement), состоящее в измельчении расчетной сетки вокруг особенности исследуемой функции. В рамках работы рассматривается один из вариантов H-уточнения, адаптивное измельчение сетки (AMR). Различные прикладные пакеты реализуют технологию динамической адаптации разными методами. Такие пакеты, как Chombo [1, 2], SAMRAI [3, 4], AMReX [5], используют блочную модель адаптации, предложенную в [6]. Особенностью этого подхода является измельчение сетки блоками, каждый из которых представляет собой небольшую регулярную сетку. Другие пакеты, такие как p4est, t8code, OpenFoam [7–9], используют листовую модель адаптации. Этот подход сопряжен с необходимостью обработки каждого сеточного элемента отдельно, что приводит к значительному увеличению накладных расходов на выполнение операции адаптации, однако позволяет сократить количество сеточных элементов, участвующих в расчете.

Особенности реализации численных методов на динамически адаптивных сетках накладывают дополнительные ограничения на структуру сетки, например в работе [10] описана сложность построения аппроксимации на границе уровней измельчения.

Операция динамической адаптации в листовой модели является затратной по вычислительным ресурсам: время, затрачиваемое на адаптацию, достигает 10% от общего времени выполнения расчета [8]. Для сокращения общего времени счета операция адаптации выполняется не на каждом шаге расчета, а через некоторые априори заданные интервалы. Чтобы обеспечить нахождение особенности исследуемой

2025, **26** (4), 410–421. doi 10.26089/NumMet.v26r427

функции в области максимального измельчения расчетной сетки, измельчение осуществляется не только в области выполнения функционального критерия адаптации, но и на некотором заранее заданном расстоянии от нее, формируя буферный слой ячеек максимального уровня измельчения между областью выполнения функционального критерия и областями более грубой сетки, а также между ячейками различных уровней измельчения. В [9] описан алгоритм построения такого слоя, основанный на выставлении целевого уровня измельчения для каждой ячейки.

В работе подробно рассматривается реализация алгоритма построения буферного слоя ячеек для листовой модели динамической адаптации на распределенной вычислительной системе. Алгоритм предполагает построение буферного слоя итеративным обходом сеточных элементов, начиная с границы области, размеченной функциональным критерием для измельчения.

В разделе 2 приводятся основные определения, используемые в работе. В разделах 3 и 4 приводятся последовательный и параллельные варианты алгоритма соответственно. В разделе 5 описаны особенности выполнения алгоритма на различных сетках на распределенной вычислительной системе. В разделе 6 приводятся оценки алгоритмической сложности алгоритма. Раздел 7 содержит описание модельной физической задачи, на которой проводилось исследование алгоритма. В разделе 8 приведены результаты численного моделирования.

**2. Определения.** Для дальнейшего изложения введем некоторые ключевые понятия, необходимые для описания работы с адаптивными сетками.

Здесь и далее будем полагать, что прямоугольная расчетная область аппроксимирована регулярной декартовой сеткой размером $N_1 \times N_2 \times N_3$ элементов. Для определенности положим $N_1 \geqslant N_2 \geqslant N_3$, $N$ — общее количество ячеек сетки. Максимальный уровень измельчения обозначим $dl_{\max}$, уровень измельчения ячейки сетки обозначим $dl_{\mathrm{c}}$, $0 \leqslant dl_{\mathrm{c}} \leqslant dl_{\max}$.

При распараллеливании положим $p$ — количество MPI-процессов, $n_i$ — количество элементов домена, принадлежащего процессу с номером $i$.

Под смежностью ячеек сетки понимается наличие у двух ячеек сетки хотя бы одного общего элемента: вершины, ребра, грани.

Функциональный критерий определяет необходимость измельчения/огрубления ячейки сетки на основании значений сеточных функций, обозначим его $f_{\mathrm{c}}$. Под выполнением критерия будем понимать ситуацию, при которой функциональный критерий определил необходимость измельчения ячейки сетки.

Структурный критерий отвечает за формирование буферного слоя и определяет, какие ячейки, для которых не выполнился функциональный критерий, также необходимо измельчить. Ширина полосы структурного критерия обозначается $l_{\mathrm{struct}}$. Для удобства введем величину $l_{\mathrm{st}} = \left\lceil \dfrac{l_{\mathrm{struct}}}{2} \right\rceil$. Значение структурного критерия в ячейке $c$ обозначается $l_c$ и определяется на границе областей, которым согласно функциональному критерию требуется измельчение. Будем для определенности полагать, что $l_c > 0$ означает необходимость измельчения ячейки, иначе — необходимость огрубления. Перед началом выполнения алгоритма $l_c = 0$ во всех ячейках сетки.

Операции, выполняемые при адаптации, — дробление и огрубление (слияние). Дробление — разделение сеточного элемента на несколько элементов меньшего размера, а огрубление — обратная операция.

Будем называть нераздробленные ячейки листовыми, а раздробленные — нелистовыми (родительскими) соответственно.

За один вызов операции адаптации возможно изменение сеточных элементов не более чем на один уровень.

Для определенности в качестве библиотеки декомпозиции расчетной сетки во время балансировки нагрузки используется пакет ParMETIS [11].

**3. Алгоритм применения структурного критерия.** Согласно определению, приведенному в разделе 2, структурный критерий при динамической адаптации отвечает за изменение (измельчение или огрубление) сеточных элементов, в которых не выполнился функциональный критерий.

Структурный критерий формируется во всех ячейках сетки. Отправной точкой выполнения алгоритма является список ячеек максимального на момент адаптации уровня измельчения, для которых хотя бы в одной смежной ячейке выполнился функциональный критерий. Для листовых ячеек, в которых функциональный критерий $f_{\mathrm{c}} = 1$ определил необходимость измельчения (или $f_{\mathrm{c}} = 0$, т.е. отсутствие необходимости изменений), значение структурного критерия $l_c = 1$, если $dl_{\mathrm{c}} = dl_{\max}$, или $l_c = l_{\mathrm{st}} + 1$ во

2025, **26** (4), 410–421. doi 10.26089/NumMet.v26r427

всех остальных случаях. Значение структурного критерия вычисляется по формуле $l_c = l_{\text{nb}} - 1$ и присваивается ячейке, если $l_{\text{nb}} - l_c > 1$, где $l_{\text{nb}}$ — максимальное значение структурного критерия по всем смежным элементам. Добавляя в список всех соседей рассмотренной ячейки и удаляя ее саму, на каждом шаге получается разметить все ячейки одного уровня измельчения. Обход ячеек сетки по уровням измельчения, начиная с самой подробной сетки, с применением описанного выше алгоритма позволяет разметить все необходимые для измельчения ячейки. Алгоритм завершает работу в тот момент, когда $l_c = 1$, $l_{\text{nb}} = 0$, ячейка имеет минимальный уровень измельчения.

Для нелистовых элементов выполняется дополнительная операция получения значения структурного критерия на основе листовых элементов. Если значение структурного критерия хотя бы в одном из потомков не равно 0, то ищется значение $l_{\max c}$ — максимальное значение величины $l_c$ среди всех потомков ячейки, тогда $l_c = l_{\text{st}} + \left\lceil \dfrac{l_{\max c}}{2} \right\rceil$.

Рассмотрим применение структурного критерия на примере. На рис. 1 a показана вся сетка (значение $l_{\text{struct}} = 2$), сплошным цветом выделена область, в которой выполнился функциональный критерий. В этих ячейках значение структурного критерия $l_c = 1$. Для ячеек максимального уровня измельчения (рис. 1 b) алгоритм выполняет однократный обход сеточных элементов и останавливается, так как выполнено условие остановки. На следующем шаге выполняется рассмотрение ячеек предыдущего уровня (рис. 1 c). Сначала для родительских ячеек, в листьях которых $l_c > 0$ (отмечены левой штриховкой), определяется значение структурного критерия, в них $l_c = 2$. На основании этих ячеек формируется следующий слой ячеек, в которых таким же образом выполняется разметка (отмечены правой штриховкой). После завершения процедуры для ячеек первого уровня алгоритм повторно применяется для ячеек начальной регулярной сетки (рис. 1 d). В ячейках, отмеченных гексаэдральной штриховкой, значение $l_c = 3$. После разметки регулярной сетки алгоритм завершает выполнение.

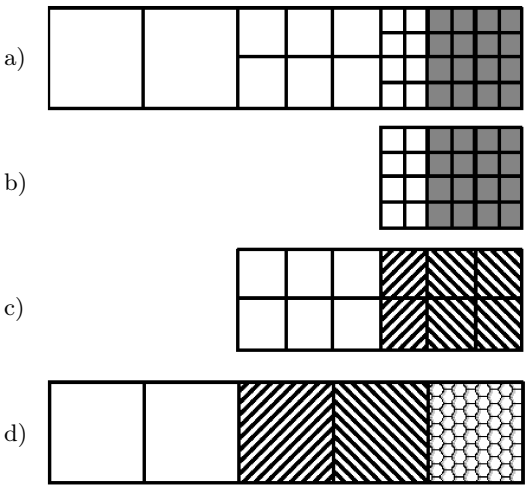


Рис. 1. Структура сетки с разделением по уровням измельчения (подробное описание в разделе 3): a) структура сетки до преобразований; b) листовые ячейки деревьев с ячейками максимального уровня измельчения; c) ячейки первого уровня измельчения; d) ячейки регулярной сетки

Fig. 1. Mesh structure divided by refinement levels (detailed description in section 3): a) mesh structure of leaf elements; b) leaf cells of trees with cells of the maximum refinement level; c) cells of the first refinement level; d) cells of a regular mesh

Алгоритм справедлив для случая, когда за одну операцию адаптации изменение уровня измельчения элементов сетки производится не более чем на единицу.

**4. Последовательная реализация структурного критерия.** Реализация структурного критерия подразумевает выполнение двух операций: продвижения волны на одну ячейку и обход сеточных элементов.

Перед началом выполнения алгоритма формируется массив массивов `nbs`, содержащий все ячейки, смежные с рассматриваемой. Размер внешнего массива составляет 27 элементов (все возможные направления смежности для гексаэдральной ячейки, включая ее саму), внутренние массивы содержат от одного до четырех элементов, содержащих ссылки на листовые ячейки. В качестве входных данных алгоритм принимает номер текущего сеточного элемента `c`, ссылку на структуру `nbs` и ссылку на сеточную структуру `msh`.

Реализация алгоритма S определения значения $l_c$ в сеточном элементе на языке C++ представлена в листинге 1. Принадлежность ячейки `nb` расчетной области проверяется в функции `check_correct_nb`. Эта же функция производит сравнение уровней измельчения сетки.

Рассмотренный выше алгоритм позволяет применить структурный критерий в рамках одной ячейки сетки. Для каждой из смежных ячеек структурный критерий устанавливается равным значению структурного критерия в текущей ячейке, уменьшенным на единицу, но не менее 0.

2025, **26** (4), 410–421. doi 10.26089/NumMet.v26r427

Листинг 1. Алгоритм применения структурного критерия в рамках одной ячейки (алгоритм S)
Listing 1. Algorithm for applying the structural criterion within a single cell (algorithm S)

```
// Traverse all cells adjacent to the current one
for(int i = 0; i < n_nb; i++)
{
    // Checking for the existence of a neighboring cell
    if(nbs[i][0] < 0)
        continue;
    // If the current cell is on the boundary of refinement levels,
    // perform a call to the parent of adjacent cells
    if(nbs[i].size() > 1)
        nb = msh.getcell(nbs[i][0]).getparent();
    else
        nb = nbs[i][0];

    // Checking the belonging of an adjacent cell to the calculation domain and
    // the equality of the grinding levels
    if(!check_correct_nb(c, nb, msh))
        continue;

    l_nb = msh.getcell(nb).get_l_range();

    // Condition for completion of the wave passage
    if((!msh.getcell(nb).get_div_level()) \
                && (l_c == 1) \
                && (l_nb == 0))
        continue;

    // Application of structural priming
    if(l_nb < l_c)
    {
        msh.getcell(nb).set_l_range(l_c - 1);
        if(!msh.getcell(nb).ifmarked())
        {
            msh.getcell(nb).mark();
            divlevel_bnd.push_back(nb);
        }
    }
}
```

Выполнение структурного критерия начинается с ячеек, в которых выполнился функциональный критерий, и обладающих хотя бы одной смежной ячейкой, в которой не выполнился функциональный критерий. Из этих ячеек формируется список `dl_l_bnd`.

Перед выполнением алгоритма все сеточные элементы получают значение структурного критерия:

1) $l_c = 0$, если в ячейке не выполнился функциональный критерий;
2) $l_c = 1$ для ячеек максимального уровня измельчения, в которых $f_c \geqslant 0$;
3) $l_c = l_{st} + 1$ в листовых ячейках, в которых $f_c \geqslant 0$ и $dl_c < dl_{max}$.

Описанный в предыдущем разделе алгоритм применения структурного критерия в рамках одной ячейки обрабатывает только ячейки одного уровня измельчения. Здесь и далее под получением соседей ячейки будем понимать всегда получение ячеек того же или меньшего уровней измельчения, что и рассматриваемая ячейка.

Один шаг прохождения волны структурного критерия состоит из двух этапов:

- На подготовительном этапе выполняется проход по всем ячейкам и из множества ячеек, для которых не выполнился функциональный критерий, выбираются такие, для которых существует хотя бы

2025, **26** (4), 410–421. doi 10.26089/NumMet.v26r427

одна смежная ячейка, отмеченная функциональным критерием к измельчению; выбранные ячейки помещаются в список `dl_l_bnd`.

- Во время обхода ко всем элементам `dl_l_bnd` применяется алгоритм S. При формировании структуры `nbs` на границе уровней измельчения, если смежная листовая ячейка имеет уровень дробления меньший, чем у текущей, то она не рассматривается. В противном случае берется значение структурного критерия из родительской ячейки, породившей смежную листовую. Алгоритм выполняется для каждого уровня измельчения последовательно, начиная с ячеек наибольшего уровня измельчения.

Выполнение алгоритма продолжается до тех пор, пока список `dl_l_bnd` содержит хотя бы один элемент. Реализация на языке C++ алгоритма обхода C представлена в листинге 2.

В силу того, что значение $l_c$ в рамках обхода ячеек (цикл while в листинге 2) может только уменьшаться и строго ограничено значениями $l_{\text{struct}}$ и 0 сверху и снизу соответственно, алгоритм выполняется за конечное время. Количество ячеек в списке `dl_l_bnd` варьируется от 0 до некоторого $M < N$. Количество итераций выполнения алгоритма зависит от $l_{\text{struct}}$ и от максимального уровня измельчения $dl_{\max}$. Количество вызовов алгоритма составляет $l_{\text{st}}(dl_{\max} + 1) + 1$. Первый этап алгоритма выполняется за один проход по сетке, вычислительная сложность этапа составляет $O(N)$. Второй этап состоит в однократном проходе по списку `dl_l_bnd`, и его алгоритмическая сложность составляет $O(M)$. Общее количество операций, необходимых для выполнения алгоритма, составляет $N + M$, вычислительная сложность $O(N)$.

Алгоритм структурного критерия обходит область всегда полностью, в том числе когда в результате обхода перестроение сетки не потребуется. Алгоритм не выполнятся только в случае, если никакая ячейка не была отмечена функциональным критерием.

Следует отметить, что величина $M$ существенно зависит от геометрии адаптируемой области и может достигать 96.3% от общего количества сеточных элементов в вырожденном случае.

Листинг 2. Алгоритм прохождения волны структурного критерия (алгоритм C)
Listing 2. Algorithm of passing the wave of the structural criterion (algorithm C)

```
// Cycle through the levels of refinement, starting with the most detailed grid
for(int i = max_div_level; i >= 0; i--)
{
    // Determining the value of a structural criterion in parent elements
    // based on information from leaves
    for(int j = 0; j < n_parent[i]; j++)
        set_l_parent(prnt[i][j], msh);

    // Formation of the wave front of the structural criterion
    form_wave(i, dl_l_bnd, msh);

    while(dl_l_bnd.size())
    {
        cellNumber c = dl_l_bnd.front();

        // Cycle termination condition
        if(msh.getcell(c).get_l_range() == 0)
        {
            dl_l_bnd.pop_front();
            continue;
        }

        msh.get_nb(c, nbs);
        l_to_cell(c, nbs, dl_l_bnd, msh);
        dl_l_bnd.pop_front();
    }
}
```

2025, **26** (4), 410–421. doi 10.26089/NumMet.v26r427

**5. Реализация алгоритма на распределенной вычислительной системе.** Выполнение вышеописанного алгоритма на распределенной вычислительной системе затруднено в силу невозможности априорного определения областей, размечаемых для измельчения, без привязки к конкретной задаче.

На рис. 2 представлен еще один вариант вырожденного случая. Залитая цветом область соответствует области выполнения функционального критерия. Величина $h_{\text{domain}}$ — ширина в ячейках домена 2 процесса в конкретном месте. В случае если $l_{\text{struct}} > h_{\text{domain}}$, для корректной разметки доменов функциональным критерием потребуется как минимум три итерации глобальных обменов:

1) после разметки первого процесса до границы со вторым процессом;
2) разметка второго процесса в показанной области, в том числе до границы с первым процессом;
3) повторная разметка первого процесса.

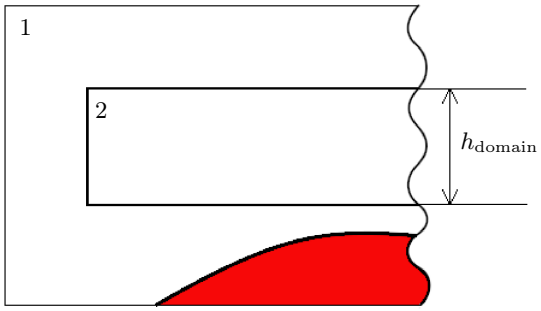


Рис. 2. Пример вырожденного случая

Fig. 2. Example of a degenerate case

Описанный выше алгоритм C применим в рамках одного домена. Количество ячеек, которые необходимо измельчить для соблюдения условия структурного критерия, может превышать ширину полей обменов между MPI-процессами (рис. 2). Следовательно, необходимо реализовать переход волны структурного критерия между MPI-процессами.

Получение значений $l_c$ для фиктивных ячеек, использующихся при межпроцессных обменах, сопряжено с необходимостью выполнения операции обмена данными между MPI-процессами.

Очередность MPI-обменов может влиять на их количество, необходимое для завершения разметки ячеек сетки структурным критерием. Рассмотрим следующий простой пример. Пусть некоторая сетка разбита на три домена между тремя MPI-процессами. Между ними будет выполнено два независимых попарных обмена: между процессами с номерами 0 и 1, 1 и 2. В такой последовательности обменов между процессами 0 и 2 не будет. Пусть функциональный критерий выполнился для некоторых ячеек на MPI-процессе 2. Тогда за одну операцию обмена волна структурного критерия сможет попасть только на процесс 1, но не сможет попасть на процесс 0. Следовательно, для полного прохождения волны структурного критерия по всей сетке потребуются две глобальные операции обмена.

В силу того, что волна структурного критерия может проходить по разным процессам независимо, необходима операция проверки завершения прохождения волны. Количество операций будет зависеть от попадания в область фиктивных ячеек элементов, инцидентных по узлу и ребру. Таким образом, вычислительная сложность алгоритма зависит от декомпозиции на домены и ширины слоя фиктивных ячеек, при этом общее количество операций глобальных обменов данными не может быть меньше двух.

Параллельная реализация алгоритма состоит из трех этапов:

1. Описанный в предыдущем разделе алгоритм применяется к каждому домену в отдельности, при этом виртуальные ячейки не учитываются при разметке критерием;
2. Операция глобального обмена данными обновляет значения структурного критерия в фиктивных ячейках, формируется новый список `dl_l_bnd` на основании этих элементов;
3. Выполняется операция глобального обмена информацией о наличии изменений в значениях структурного критерия. Если значение структурного критерия не изменялось ни в одной ячейке сетки, алгоритм завершается.

**6. Алгоритмическая сложность.** Рассмотрим следующий предельный случай: разбиение сетки, когда никакие две смежные ячейки не принадлежат одному MPI-процессу. Тогда за одну итерацию волна продвинется на одну ячейку по грани, ребру и узлу, после чего возникнет необходимость операции глобального обмена данными. Количество обменов, требуемое для завершения выполнения алгоритма в таком случае оказывается $O(\max(N_1, p))$.

Вторым предельным случаем станет разбиение сетки на домены так, что каждый процесс обменивается только с двумя процессами: процесс с номером $i$ обменивается только с процессами $i+1$ и $i-1$. Волна структурного критерия распространяется с процесса 0 и требует разметки всей сетки. В таком

2025, **26** (4), 410–421. doi 10.26089/NumMet.v26r427

случае для полной разметки сетки потребуется $O(n_i \times p)$ операций, количество обменов будет составлять $O(p)$, а сама операция разметки фактически будет выполнятся последовательно.

На практике количество сеточных элементов, размечаемых структурным критерием, оказывается меньше размера сетки и составляет порядка 10% от общего числа элементов. Часть алгоритмов разбиения на домены стараются сохранить области односвязными (например, [11, 12]). При использовании таких подходов количество итераций выполнения не превышает 2–3. В силу того, что волна формирования буферного слоя элементов будет проходить по некоторым конфигурациям разбиения на домены последовательно, время выполнения будет зависеть от геометрии доменов.

**7. Вычислительный эксперимент.** Для демонстрации работы алгоритма в качестве модельной рассматривается задача о взаимодействии ударной волны с вихрем (shock-vortex interaction, SVI), постановка задачи взята из [13].

Базовая физико-математическая модель, использующаяся для моделирования обозначенной выше задачи, определяется системой уравнений Эйлера для сжимаемого нетеплопроводного газа:

$$\begin{cases} \dfrac{\partial}{\partial t}\rho + \dfrac{\partial}{\partial x_i}(\rho v_i) = 0, \\ \dfrac{\partial}{\partial t}(\rho v_i) + \dfrac{\partial}{\partial x_j}(\rho v_i v_j + p\delta_{ij}) = 0, \\ \dfrac{\partial}{\partial t}(\rho E) + \dfrac{\partial}{\partial x_i}(\rho v_i E + p v_i) = 0, \\ p = p(\rho, \varepsilon), \end{cases} \tag{1}$$

где $E = \varepsilon + \dfrac{1}{2} v_k v_k$ — полная удельная энергия, $\varepsilon$ — удельная внутренняя энергия, $p$ — давление, $\rho$ — плотность, $v_i$ — компоненты скорости.

Задача решается в системе отсчета, связанной с фронтом ударной волны, т.е. фронт ударной волны остается неподвижным, испытывая небольшие колебания в результате прохождения через него вихря. На левой границе задаются условия сверхзвукового втекающего потока, на правой — дозвукового вытекающего потока.

Адаптация выполнялась до максимального уровня измельчения $l_{\max} = 2$, базовая сетка имела размерность $200 \times 100 \times 4$ ячеек, результирующий размер сетки достигал значения 873715 ячеек. Расчет псевдодвумерный, некоторое количество ячеек вдоль оси $z$ обусловлено особенностями работы библиотеки для работы с трехмерным AMR octreemesh. Ширина полосы структурного критерия $l_{\text{struct}} = 4$, адаптация выполнялась каждые 20 шагов по времени.

Функциональный критерий был определен как характерный масштаб изменения плотности $L_\rho = \left(|\nabla \ln \rho|\right)^{-1}$, для которого использовались пороговые значения $L_\rho^{\text{th},1} = 0.1$ и $L_\rho^{\text{th},2} = 1$. При $L_\rho < L_\rho^{\text{th},1}$ ячейки помечались для измельчения ($f_c = 1$), при $L_\rho > L_\rho^{\text{th},2}$ — для объединения ($f_c = 0$). При промежуточных значениях текущий уровень измельчения не менялся.

Для численного решения системы уравнений (1) используется схема второго порядка аппроксимации по времени и по пространству. Для повышения порядка аппроксимации по пространству используется TVD-схема [14] с линейной интерполяцией вектора физических переменных на грани ячеек. Поток через грани ячеек вычисляется в соответствии с приближенным решением задачи о распаде разрыва [15]. Численное интегрирование по времени осуществляется схемой “предиктор–корректор”.

**8. Результаты.** Численное моделирование выполнялось с использованием 20 MPI-процессов на одном узле на вычислительном кластере K-60 [16].

На рис. 3 показано состояние сетки в момент перед столкновением вихря и ударной волны. Структурный критерий отработал таким образом, что между фронтом волны и границей вихря сначала возникает область ячеек первого уровня измельчения, а после, как показано на рис. 4, полностью заполняется ячейками максимального уровня измельчения.

В процессе развития неустойчивости функциональный критерий начинает выполняться в различных зонах расчетной области. На момент завершения расчета (рис. 5) можно видеть достаточно сложные структуры, образовавшиеся за фронтом ударной волны и вокруг вихря. Видно, что предложенный алгоритм реализации структурного критерия позволяет строить буферный слой ячеек в подобных случаях.

2025, **26** (4), 410–421. doi 10.26089/NumMet.v26r427

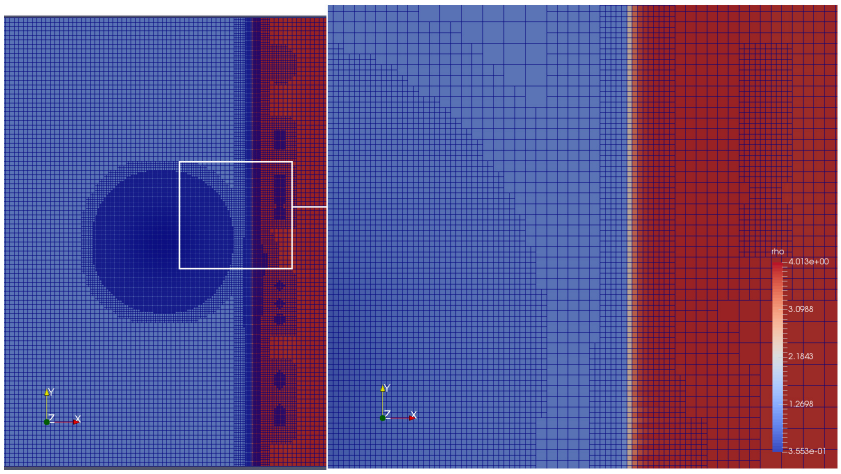


Рис. 3. Структура сетки перед столкновением вихря и ударной волны

Fig. 3. Mesh structure before the collision of a vortex and a shock wave

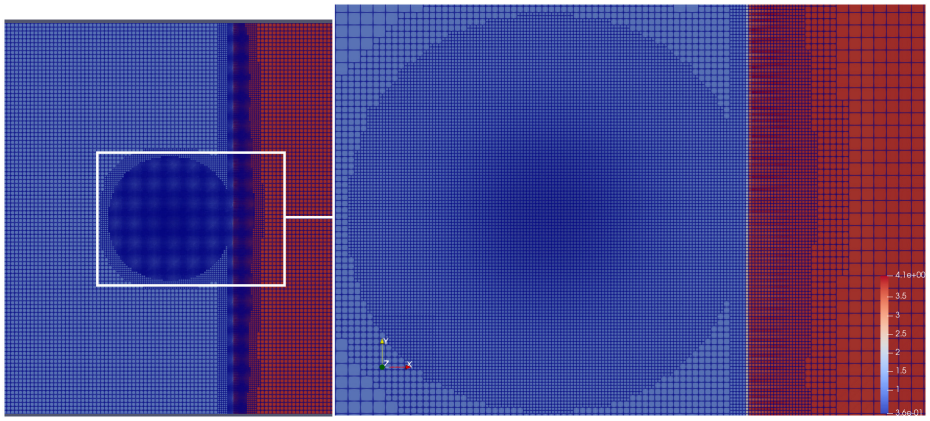


Рис. 4. Структура сетки в момент столкновения

Fig. 4. Mesh structure in the moment of collision

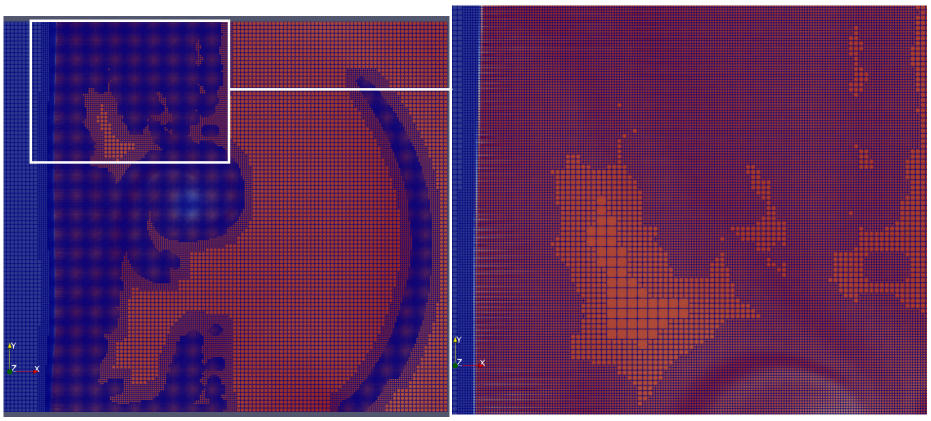

Рис. 5. Структура сетки в конце расчета

Fig. 5. Mesh structure at the end of calculation

2025, **26** (4), 410–421. doi 10.26089/NumMet.v26r427

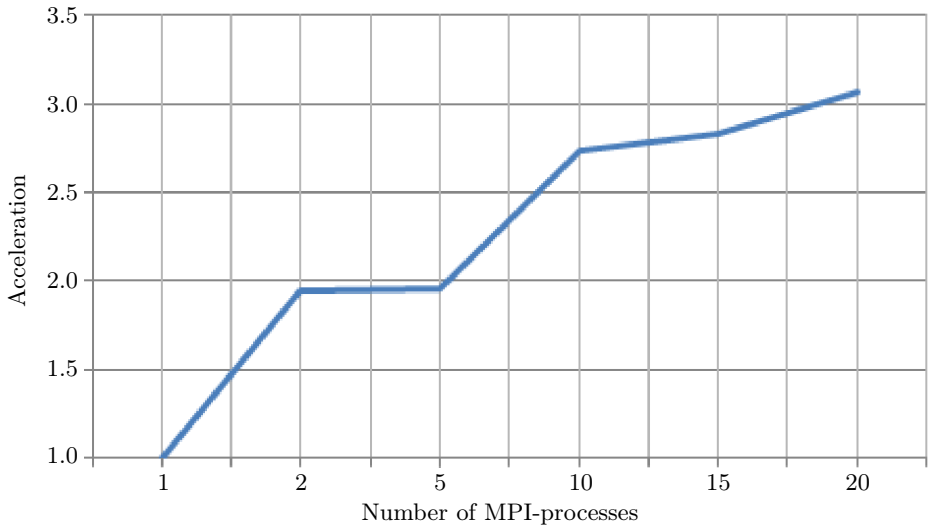


Рис. 6. Зависимость ускорения от числа MPI-процессов

Fig. 6. Dependence of acceleration on the number of MPI processes

Расчет проводился до $t = 0.5$, максимальное количество глобальных обменов при разметке структурным критерием составило 3.

Время выполнения алгоритма на рассматриваемой сетке не превышало 2.39 с в последовательном случае, 0.78 с на 20 MPI-процессах. На рис. 6 показана зависимость ускорения алгоритма от числа MPI-процессов. Время выполнения обусловлено количеством сеточных элементов, размечаемых структурным критерием. Плохая масштабируемость алгоритма обусловлена тем, что декомпозиция сетки на домены не учитывает количество элементов, размечаемых структурным критерием на каждом MPI-процессе.

**9. Заключение.** В библиотеках динамической адаптации расчетной сетки для уменьшения ошибок дискретизации между различными уровнями измельчения строится буферный слой. Ширина буферного слоя зависит от конкретного численного метода.

Рассмотренный алгоритм позволяет формировать буферный слой ячеек в областях достаточно сложной геометрической формы. Недостатком алгоритма является его слабая масштабируемость.

Разметка выполняется с учетом области выполнения функционального критерия последовательным обходом сеточных элементов. Начиная с самой подробной сетки, каждый соседний элемент отмечается значением $l_c$ на единицу меньшим, чем текущий, если значение структурного критерия в соседнем элементе меньше, чем в текущем более чем на 1, вплоть до $l_c = 0$. Алгоритм завершается, когда оказывается размечена наиболее грубая сетка.

В последовательном случае оценка вычислительной сложности последовательного алгоритма составляет $O(N_1)$. При распараллеливании количество сеансов глобальных обменов в худшем случае $\max(N_1, p)$. Показано, что на количество сеансов глобальных обменов данными влияет очередность обменов данными и геометрические особенности доменов.

Рассмотренные алгоритмы реализованы на языке C++ в пакете работы с динамически адаптивными сетками octreemesh, для балансировки нагрузки использовался пакет ParMETIS [11].

Для апробации алгоритма было выполнено численное моделирование задачи о взаимодействии ударной волны с вихрем (SVI). При расчете количество сеансов глобального обмена данными не превышало 3 для 20 MPI-процессов на сетке размером $10^6$ элементов. Небольшое количество обменов связано со свойствами алгоритмов декомпозиции сеток. Максимальное ускорение, которого удалось достичь, было трехкратным, время выполнения составляло 0.78 с.

2025, **26** (4), 410–421. doi 10.26089/NumMet.v26r427

### Список литературы


1. Chombo — Software for Adaptive Solutions of Partial Differential Equations — Chombo — Berkeley Lab Commons. http://chombo.lbl.gov/. (Дата обращения: 23 сентября 2025).
2. *Adams M., Colella P., Graves D.T., Johnson J.N., Keen N.D., Ligocki T.J., et al.* Chombo Software Package for AMR Applications — Design Document // Lawrence Berkeley National Laboratory Technical Report LBNL-6616E. 2015. https://escholarship.org/uc/item/5cs5d1sq. (Дата обращения: 23 сентября 2025).
3. SAMRAI | Computing. https://computing.llnl.gov/projects/samrai. (Дата обращения: 23 сентября 2025).
4. *Hornung R.D., Kohn S.R.* Managing application complexity in the SAMRAI object-oriented framework // Concurrency and Computation: Practice and Experience. 2002. **14**, 347–368. doi 10.1002/cpe.652.
5. *Zhang W., Almgren A., Beckner V., Bell J., et al.* AMReX: a framework for block-structured adaptive mesh refinement // Journal of Open Source Software. 2019. **4** (37), 1370. doi 10.21105/joss.01370.
6. *Berger M.J., Colella P.* Local adaptive mesh refinement for shock hydrodynamics // Journal of Computational Physics. 1989. **82**, 64–84. doi 10.1016/0021-9991(89)90035-1.
7. *Burstedde C., Wilcox L.C., Ghattas O.* p4est: Scalable Algorithms for Parallel Adaptive Mesh Refinement on Forests of Octrees // SIAM Journal on Scientific Computing. 2011. **33** (3), 1103–1133. doi 10.1137/100791634.
8. *Holke J., Burstedde C., Knapp D., Dreyer L., Elsweijer S., Ünlü V., Markert J., Lilikakis I., Böing N., Ponnusamy P., Basermann A.* t8code v. 1.0 — Modular Adaptive Mesh Refinement in the Exascale Era // SIAM International Meshing Round Table. Amsterdam, Niederlande, March 6–9, 2023. https://internationalmeshingroundtable.com/assets/research-notes/imr31/2017-comp.pdf. (Дата обращения: 23 сентября 2025).
9. *Rettenmaier D., Deising D., Ouedraogo Y., et al.* Load balanced 2D and 3D adaptive mesh refinement in OpenFOAM // SoftwareX. 2019. **10**, 100317. doi 10.1016/j.softx.2019.100317.
10. *Ольховская О.Г.* Проекционно-сеточные схемы для аппроксимации уравнений в частных производных второго порядка на нерегулярных сетках. Препринт № 226. М.: Ин-т Прикл. Матем. им. Келдыша, 2018. doi 10.20948/prepr-2018-226.
11. Karypis G. *METIS and ParMETIS. Encyclopedia of Parallel Computing (eds. Padua D. eds).* Boston: Springer, 2011. 1117–1124. doi 10.1007/978-0-387-09766-4_500.
12. *Hendrickson B., Kolda T.G.* Graph partitioning models for parallel computing // Parallel Computing. 2000. **26** (12), 1519–1534. doi 10.1016/S0167-8191(00)00048-X.
13. Institute of Applied Mathematics - V&V_2022_SVI_test_problem.pdf. (“Взаимодействие ударной волны с вихрем: тестовая задача для методов сквозного счета”). https://ceaa.imamod.ru/2022/files/V&V_2022_SVI_test_problem.pdf. (Дата обращения: 23 сентября 2025).
14. *Тишкин В.Ф., Никишин В.В., Попов И.В., Фаворский А.П.* Разностные схемы трехмерной газовой динамики для задачи о развитии неустойчивости Рихтмайера–Мешкова // Матем. моделирование. 1995. **7**, № 5, 15–25.
15. *Shen Z., Yan W., Yuan G.* A robust HLLC-type Riemann solver for strong shock // J. Comput. Phys. 2016. **309**, 185–206. doi 10.1016/j.jcp.2016.01.001.
16. ЦКП ИПМ РАН. https://ckp.kiam.ru. (Дата обращения: 23 сентября 2025).


*Получена* 7 августа 2025 г. | *Принята* 22 сентября 2025 г. | *Опубликована* 9 октября 2025 г.

### Информация об авторах

*Сергей Константинович Григорьев* — мл. науч. сотр.; Институт прикладной математики имени М. В. Келдыша (ИПМ РАН), Миусская пл., 4, 125047, Москва, Российская Федерация.

*Павел Александрович Кучугов* — к.ф.-м.н., ст. науч. сотр.; Институт прикладной математики имени М. В. Келдыша (ИПМ РАН), Миусская пл., 4, 125047, Москва, Российская Федерация.

### References


1. Chombo — Software for Adaptive Solutions of Partial Differential Equations — Chombo — Berkeley Lab Commons. http://chombo.lbl.gov/. Cited September 23, 2025.
2. M. Adams, P. Colella, D. T. Graves, J. N. Johnson, N. D. Keen, T. J. Ligocki, et al., “Chombo Software Package for AMR Applications — Design Document”, Lawrence Berkeley National Laboratory Technical Report LBNL-6616E, January 9, 2015. https://escholarship.org/uc/item/5cs5d1sq. Cited September 23, 2025.

2025, **26** (4), 410–421. doi 10.26089/NumMet.v26r427


3. SAMRAI | Computing. https://computing.llnl.gov/projects/samrai. Cited September 23, 2025.
4. R. D. Hornung and S. R. Kohn, "Managing application complexity in the SAMRAI object-oriented framework," Concurrency and Computat.: Pract. Exper. **14** (5), 347–368 (2002). doi 10.1002/cpe.652.
5. W. Zhang, A. Almgren, V. Beckner, J. Bell, et al., "AMReX: a framework for block-structured adaptive mesh refinement," Journal of Open Source Software **4** (37), 1370. (2019). doi 10.21105/joss.01370.
6. M. J. Berger, P. Colella, "Local adaptive mesh refinement for shock hydrodynamics," Journal of Computational Physics **82**, 64–84 (1989). doi 10.1016/0021-9991(89)90035-1.
7. C. Burstedde, L. C. Wilcox, and O. Ghattas, "p4est: Scalable Algorithms for Parallel Adaptive Mesh Refinement on Forests of Octrees," SIAM J. Sci. Comput. **33** (3), 1103–1133 (2011). doi 10.1137/100791634.
8. J. Holke, C. Burstedde, D. Knapp, L. Dreyer, S. Elsweijer, V. Ünlü, J. Markert, I. Lilikakis, N. Böing, P. Ponnusamy, A. Basermann, "t8code v. 1.0 — Modular Adaptive Mesh Refinement in the Exascale Era," in *SIAM International Meshing Round Table. Amsterdam, Niederlande, March 6–9, 2023.* https://internationalmeshingroundtable.com/assets/research-notes/imr31/2017-comp.pdf. Cited September 23, 2025.
9. D. Rettenmaier, D. Deising, Y. Ouedraogo, et al., "Load balanced 2D and 3D adaptive mesh refinement in OpenFOAM," SoftwareX **10**, 100317 (2019). doi 10.1016/j.softx.2019.100317.
10. O. G. Olkhovskaya, *Grid-projection schemes for approximation of the second order partial differential equations on irregular computational meshes*, Preprint No. 226 (Keldysh Institute of Applied Mathematics, Moscow, 2018) doi 10.20948/prepr-2018-226. [in Russian].
11. G. Karypis, "METIS and ParMETIS," In: Encyclopedia of Parallel Computing (ed. D. Padua). (Springer US, Boston, MA), pp. 1117–1124. doi 10.1007/978-0-387-09766-4_500.
12. B. Hendrickson, T. G. Kolda, "Graph partitioning models for parallel computing," Parallel Computing **26** (12), 1519–1534 (2000). doi 10.1016/S0167-8191(00)00048-X.
13. Institute of Applied Mathematics - V&V_2022_SVI_test_problem.pdf. (Shock wave-vortex interaction: test task for direct numerical simulation methodes) [in Russian]. https://ceaa.imamod.ru/2022/files/V&V_2022_SVI_test_problem.pdf. Cited September 23, 2025.
14. V. F. Tishkin, V. V. Nikishin, I. V. Popov, A. P. Favorski, "Finite difference schemes of three-dimensional gas dynamics for the study of Richtmyer–Meshkov instability," Mat. Model. **7** (5), 15–25 (1995). [in Russian].
15. Z. Shen, W. Yan, G. Yuan, "A robust HLLC-type Riemann solver for strong shock," J. Comput. Phys. **309**, 185–206 (2016). doi 10.1016/j.jcp.2016.01.001.
16. Center for Collective Use of the Keldysh Institute of Applied Mathematics of the Russian Academy of Sciences. Hybrid supercomputer K60. [in Russian]. https://ckp.kiam.ru. Cited September 23, 2025.



| *Received* | *Accepted* | *Published* |
|---|---|---|
| *August 7, 2025* | *September 22, 2025* | *October 9, 2025* |


### Information about the authors


*Sergey K. Grigoriev* — Junior Scientist; Keldysh Institute of Applied Mathematics of RAS, Miusskaya ploshchad', 4, 125047, Moscow, Russia.

*Pavel A. Kuchugov* — Ph.D., Senior Scientist; Keldysh Institute of Applied Mathematics of RAS, Miusskaya ploshchad', 4, 125047, Moscow, Russia.