Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра суперкомпьютеров и квантовой информатики

Соколова Ксения Максимовна

Разработка гибридных параллельных алгоритмов и программ для различных стратегий распределения вычислений между процессорами

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Научный руководитель:
д.ф-м.н., профессор
М.В. Якобовский

Москва, 2019

Аннотация

В данной работе проводится исследование подходов к представлению декартовых локально-сгущающихся сеток, структур данных для их представления, методов распределения таких сеток между вычислительными узлами многопроцессорной системы и алгоритмов обработки сеток внутри вычислительных узлов. На основе приведённого исследования предлагается новый подход к представлению декартовых локально-сгущающихся сеток, решающий проблему нерегулярности доступа к памяти, и эффективные алгоритмы их гибридной параллельной обработки.

Оглавление

1. Введение

1.1 Актуальность

Для моделирования физических процессов требуются большие вычислительные ресурсы, поэтому для их решения используются параллельные вычисления с использованием суперкомпьютеров. Основными критериями качества программ, решающих задачи моделирования физических процессов, являются время счета и точность получаемого решения.

Во многих задачах термо- и газодинамики, описывающих активно протекающие процессы, моделируемая область является разнородной: в некоторых местах процессы протекают быстро и амплитудно, в других плавно. При этом погрешность решения пропорциональна амплитуде изменения физической величины, т.е. области резких изменений требуют более частой сетки для сохранения такой же погрешности, однако при измельчении всей области время счета сильно возрастает, поэтому для таких задач используются сетки с локальным измельчением. Если моделируется процесс переноса, то области с наиболее амплитудными изменениями перемещаются, а значит необходимо чтобы измельчение сетки адаптировалось к ним. Таким образом повышается точность и сокращается время решения задачи.

В настоящее время для моделирования задач аэродинамическокого обтекания твердых тел активно развивается подход, при котором сетка полностью покрывает область, а не согласовывается с границами объекта, а вычисления проводятся с использованием метода погруженных границ или метода свободной границы. Для таких расчетов чаще всего используются декартовы локально-измельченные сетки.

Для эффективного решения задачи с использованием суперкомпьютера необходимо решать две основные задачи: балансировать нагрузку и эффективно использовать вычислительные узлы.

Балансировка нагрузки осуществляется с помощью декомпозиции расчетных сеток. Существует множество специальных пакетов программ, решающих задачу декомпозиции расчетной сетки, эти пакеты хорошо решают задачу в общем случае. Однако в большинстве конкретных конфигураций декомпозиция расчетной сетки является нетривиальной задачей, а значит остается актуальной задача определения наилучшей стратегии для конкретной конфигурации. При декомпозиции возникают следующие проблемы: накладные расходы

на межпроцессную коммуникацию, обработку и упаковку данных для обмена, а также встает вопрос об обработке получаемых доменов.

Существует два основных подхода к решению задачи декомпозиции: использование метода, оптимизированного для наилучшего баланса вычислительной нагрузки и наименьшей коммуникационной нагрузки и разбиение на удобные для работы домены с некоторым дисбалансом вычислительной нагрузки. Использование первого подхода может не дать предполагаемой выгоды из-за затрат на обработку неструктурированных областей. Таким образом становится актуальной задача разработки метода распределения нагрузки, сочетающего в себе хороший баланс оптимизированных доменов первого подхода и простоту работы с регулярными доменами второго.

Эффективность использования вычислительных узлов определяется регулярностью доступа к памяти, возможностью использования параллелизма на общей памяти, устройством обхода областей.

Для эффективного использования суперкомпьютера необходимо учитывать его архитектуру. В настоящее время суперкомпьютеры эволюционируют в направлении роста как количества процессоров в кластере, так и количества ядер в одном вычислительном узле, а также распространены графические ускорители и сопроцессоры. В связи с этим необходимо разрабатывать гибридные параллельные алгоритмы и программы. При сочетании параллельных вычислений на общей и распределенной памяти возникают проблемы одновременного доступа к памяти и оптимизации использования кэша, особенно заметные при масштабировании на большое количество узлов.

В данной работе предлагается исследовать возможность реализации метода декомпозиции локально-сгущающихся декартовых сеток на домены, имеющие регулярную структуру, с сохранением приемлемого баланса нагрузки, а также разработать эффективный масштабируемый алгоритм обработки полученных доменов.

1.2 Цель и задачи

Цель работы — исследование и разработка гибридных параллельных алгоритмов для различных стратегий распределения нагрузки между вычислительными узлами для различных способов представления локально-сгущающихся декартовых сеток.

Для достижения поставленной цели необходимо решить следующие задачи:

1 Предложить структуру данных для представления декартовых локально-сгущающихся сеток.

2 Разработать алгоритм декомпозиции расчетной сетки на домены в предложенном представлении и эффективный параллельный алгоритм обработки доменов вычислительными узлами.

3 Провести экспериментальное исследование эффективности предложенного решения, сравнить предложенный подход и подход, предлагаемый в библиотеке p4est.

1.3 Обзор источников

Основные понятия области построения расчетных сеток, критерии их качества и подходы к моделированию физических областей расчетными сетками исследуются в работах [6] и [7]. Основные проблемы при построении и использовании различных типов расчетных сеток описаны в работах [1], [5], [12]. Подходы к представлению декартовых локально-сгущающихся сеток в памяти описаны в [13], [21], [22]. Различные алгоритмы для вычислений и некоторые аспекты моделирования областей декартовыми локально-сгущающимися сетками представлены в [2], [3], [8], [13], [16], [17]. Широко используемый пакет программ p4est для работы с декартовыми локально-сгущающимися сетками и основные алгоритмы, используемые в нем, описаны в [20]. Обзор методов декомпозиции расчетных сеток проведен в [15]. Проблемы эффективного доступа к памяти и способы их решения описаны в [4].

1.4 Подход к решению задачи

Существующие способы представления декартовых локально-сгущающихся сеток основываются либо на послойной структуре, либо на древовидной структуре. Послойная структура является затратной по памяти и негибкой в использовании алгоритмов декомпозиции. Для древовидной структуры существует эффективный способ представления в памяти компьютера с помощью кривой Мортона. Однако данный способ стирает определенность адресации ячеек, а значит делает необходимым сохранение указателей на соседей. Это увеличивает объем используемой памяти, а значит ухудшает эффективность работы с кэшем. Также несмотря на то, что нумерация ячеек в порядке кривой Мортона является достаточно эффективной по использованию кэша, существуют нелокальные скачки. Таким образом, чтобы улучшить эффективность доступа к памяти, необходимо иметь регулярные последовательные блоки в памяти, чего можно добиться с помощью блочного хранения расчетной сетки.

Для декомпозиции расчетной сетки в данном представлении естественным образом вытекает декомпозиция по кривой Мортона, позволяющий быстро получить разбиение на домены с хорошим балансом нагрузки. Однако домены, полученные данным способом, имеют рваные границы, что также можно улучшить, если использовать разбиение блоками.

2. Обзор

В рамках обзора будут рассмотрены критерии качества расчетных сеток, структуры их описания, различные методы декомпозиции и работы с декомпозированными областями в общем случае и в случае декартовых локально-сгущающихся сеток.

2.1 Критерии качества расчетных сеток

Расчетные сетки должны дискретизировать физическое пространство или поверхность таким образом, чтобы вычислительные и физические характеристики могли быть обработаны максимально эффективно.

Лисейкин В.Д. в своей монографии [7] сформулировал и дополнил в монографии [6] следующие основные критерии качества сеток. В дальнейшем будем ссылаться на данные критерии.

1. Размер сетки и ячеек — размер сетки определяется количеством узлов в ней, размер ячейки определяется максимальной из длин ее ребер. Необходимо, чтобы при построении расчетной сетки была возможность увеличить количество ячеек, при этом так, чтобы размер ячейки стремился к нулю при стремлении количества ячеек к бесконечности. Размер ячейки влияет на точность решения.

2. Организация сетки — включает в себя описание соседей узла или ячейки. Если сетка имеет некоторую организацию узлов (ячеек), то становится возможным удобно формализовать ее обработку. Особенно важна структура для метода конечных разностей и конечных объемов.

3. Регулярность — некоторая мера отклонения ячейки от стандартной наименее деформированной, обычно характеризуется соотношением сторон, углом между гранями или объемом (площадью в двумерном случае) ячейки. Регулярность важна для однородности процесса приближения дифференциальных уравнений.

4. Гладкость — характеризуется скоростью изменений геометрических свойств последовательных ячеек. Соседние ячейки сетки должны не сильно отличаться по форме и размеру.

5. Согласованность с геометрией — означает, что расстояние между любыми точками области (границы области) и узлом сетки должно стремиться к нулю при стремлении размера сетки к бесконечности. От степени согласованности с геометрией зависит точность численного решения задачи.

6. Согласованность с решением — означает, что ячейки сетки должны сгущаться в областях с большей погрешностью аппроксимации. Согласованность с решением важна для получения точного решения и сохранения его свойств.

7. Совместимость с численными методами — означает, что расположение зон с локальным измельчением должно соответствовать свойствам численной аппроксимации уравнений. Для различных типов численный методов могут накладываться различные ограничения.

2.2 Подходы к моделированию областей расчетными сетками

Существует три основных подхода к решению задач моделирования физических процессов в областях сложной и/или изменяющейся геометрии: метод связных сеток, химерных сеток, декартовых сеток. У каждого из этих подходов существуют как плюсы, так и минусы.

2.2.1 Метод связных сеток

При использовании метода связных сеток генерируется сетка, точно описывающая область, с помощью специальных программ-генераторов сеток. Основное преимущество данного подхода состоит в том, что сетка точно описывает область и является однородной по своей структуре. Однако при изменении формы границ области или объектов внутри нее сетку необходимо перестраивать, причем на каждом вычислительном шаге, что требует больших вычислительных ресурсов. К тому же генераторы таких сеток сами по себе являются сложными программными комплексами, имеющими сложную теоретическую базу под собой.

Данный подход подробно описывает Лисейкин В.Д. в своей монографии [6]. Сетки, генерируемые при данном подходе, могут быть структурными и неструктурными. При генерации структурных сеток используется метод отражений, а для неструктурных используются другие методы.

Структурные сетки, согласованные с границей, строятся методом отображений простой геометрии на сложную геометрию области. Основными инструментами данного метода являются математические модели, численное решение которых на эталонной сетке позволяет находить значения промежуточного преобразования и, тем самым, формировать разностные сетки в физических областях. Сетки, полученные данным методом, удобны при формулировке условий, которым должны удовлетворять решения уравнений с частными производными в области, так как эти условия могут быть легко описаны в терминах отображений. При этом сгущение узлов сетки в областях большей вариации целевой функции достигается, если условие отображения обеспечивает небольшую вариацию отображения целевой функции.

Однако данный подход не может быть применен при построении сеток для областей со сложной геометрией границ, особенно если границы заданы дискретным набором точек. Несмотря на свободу выбора исходной геометрии, отображение ее на сильно нерегулярную структуру ведет к сильным искажениям ячеек, недостаточной точности и избыточности.

Неструктурные сетки позволяют расставлять узлы независимо от каких-либо координатных направлений, что дает возможность аппроксимировать приграничные зоны с большой кривизной границы и зоны больших вариаций решений без излишнего добавления точек сетки вне этих зон. Несмотря на то, что неструктурные сетки по определению допускают любую форму ячеек, обычно при их построении используются стандартные формы. Наиболее развитым и перспективным направлением является генерация тетраэдральных сеток. Методы построения неструктурных тетраэдральных сеток, покрывающих область, в основном базируются на двух: триангуляция Делоне и фронтальный метод. Оба метода связаны с большими вычислительными расходами на процедуры хранения и получения информации о ячейках, имеющих общие грани и ребра и на получение численного решения уравнений частных производных.

Суков С.А. описывает исследование широко используемых программных комплексов для генерации тетраэдральных сеток GAMBIT, ICEM и DECO [12] по набору критериев качества и приходит к выводу, что при заданном множестве сеточных узлов все пакеты более-менее соответствуют критериям качества, однако при неопределенном множестве присутствуют деформации сеточных элементов, а также превышение оптимального количества тетраэдров. Отклонение результатов от эталонных обусловлено тем, что задача оптимизации решается приближенно в целях увеличения производительности. Значит

остается открытым вопрос генерации оптимального распределения сеточных узлов и параллельной реализации существующих алгоритмов.

Таким образом, в настоящее время не существует генератора, удовлетворяющего одновременно критериям качества и критериям эффективности, необходимым для задач моделирования физических процессов в областях сложной и/или изменяющейся геометрии.

2.2.2 Метод химерных сеток

При данном подходе в расчетной области строится два уровня сеток: основная и дочерние. Основная сетка строится во всей расчетной области, включая обтекаемые тела, независимо от их расположения и формы. Дочерние строятся вблизи каждого объекта поверх основной сетки. Решение вблизи объектов получается на дочерних сетках и затем должно быть интерполировано на основную, что ведет к дополнительных накладным расходам во время счета и появлению дополнительных погрешностей.

Основными подходами к минимизации погрешностей интерполяции являются использование алгоритмов интерполяции повышенной точности и построение консервативных схем расчета, которые в свою очередь делятся на две категории: методики, где выполнение законов сохранения гарантируется за счет применения специальных схем расчета, основанных на геометрических характеристиках сеток, и методы, основанные на построении сеток без перекрытий.

Дерюгин Ю.Н. проводит анализ методик расчета на химерных сетках в [5] и выделяет следующие основные задачи: определение несчетных областей, нахождение пересечения сеток и построение интерполяционного шаблона для взаимодействия несвязанных сеточных регионов. Для каждой из этих задач имеются алгоритмы решения, однако все они ведут к достаточно большим накладным расходам.

2.2.3 Метод декартовых сеток

При использовании метода декартовых сеток вся область покрывается регулярной декартовой сеткой простейшей формы (куб, квадрат) и не является согласованной с границами, а физическая область пересекает некоторые ячейки. Таким образом возникают три типа ячеек: полностью включенные в расчетную область, полностью исключенные из расчетной области и усеченные, имеющие некоторую форму в зависимости от геометрии пересечения.

При данном подходе построение сетки является тривиальной задачей, и практически не несет накладных расходов, в отличие от метода связных сеток. К тому же нет

необходимости в интерполяции данных при перенесении между ячейками разных подобластей сеток, а значит не теряется точность решения, в отличие от метода химерных сеток. Более того, декартовы сетки хорошо подходят для реализации разностных схем согласно критерию качества расчетных сеток.

Главным недостатком подхода наложения сетки является необходимость пересечения с границ объекта с ячейками сетки. Для сложных геометрий в трехмерном случае это может приводить к существенным накладным расходам. Однако ведутся разработки эффективных алгоритмов решения данной задачи. Так, в [1] предлагается метод, приближающий границу области набором треугольников. Он основан на использовании дискретного геометрического представления объекта и анализе дуального графа расчетной сетки с распараллеливанием и оптимизацией кеширования по аналогии с современными методами компьютерной графики. Время обработки задачи пересечения при использовании данного метода в задаче с изменяющимися объектами сопоставимо со временем расчета основной задачи. Вторым подходом к решению задачи пересечения области с объектом является метод объемных долей [8], основанный на приближении границы по значениям доли объема, занимаемого областью. Данный метод требует меньшего объема памяти по сравнению с другими и позволяет реализовать алгоритмы, не требующие дополнительного обхода всей сетки для поиска пересечений.

Заметим, что декартовы сетки не удовлетворяют критерию соответствия сетки геометрии, однако в случае, например, изменяющейся геометрии расчетной области, они позволяют обойтись без перестроения сетки в ходе расчетов. Для этого используются специальные численные методы, основанные на методе погруженной границы (Immersed Boundary Method). Суть данных методов состоит в моделировании граничных условий на поверхности твердого тела введением в исходную систему уравнений специальных функций-источников или штрафных функций. Так, многие научные группы используют метод погруженных границ или его аналоги. Например, Афендиков А.Л. и Меньшов И.С. предлагают метод свободной границы [3] [8], в котором они предлагают вводить в правую часть уравнений корректирующие потоки. Заметим, что для реализации метода свободной границы используется представление геометрии в виде объемных долей.

Основным преимуществом данного подхода является приспособленность получаемых расчетных сеток к использованию конечно-разностных и конечно-объемных методов, имеющих регулярный шаблон доступа к памяти, а значит возможность их эффективной параллельной обработки на современных многоядерных вычислительных системах.

В дальнейшем в работе будут рассматриваться декартовы сетки.

2.3 Локально-сгущающиеся декартовы сетки

Согласно одному из критериев качества расчетных сеток, сетка должна быть адаптирована к решению, и эта адаптация достигается за счет локального сгущения сетки в областях большего градиента целевой функции. В данном пункте рассматриваются существующие способы представления локально-сгущающихся декартовых сеток, алгоритмы построения, адаптации и выполнения расчетов на их основе.

2.3.1 Представление расчетной сетки

Будем использовать следующие термины.

Ячейка расчетной сетки — элементарная вычислительная единица сетки.

Уровень ячейки — величина, характеризующая размер ячейки. Чем больше уровень, тем меньше размер. Ячейки одного уровня имеют одинаковые размеры.

Слой расчетной сетки — набор ячеек одинакового размера. Здесь под словом набор не имеется в виду конкретная структура данных.

Блок ячеек — набор ячеек одинакового размера, объединенных по некоторому принципу. Блок является подмножеством слоя.

Существует два основных подхода к представлению декартовой локально-сгущающейся сетки в памяти: блочно-ориентированный и ячейко-ориентированный, схематично представленные на ***Рисунок 1***.

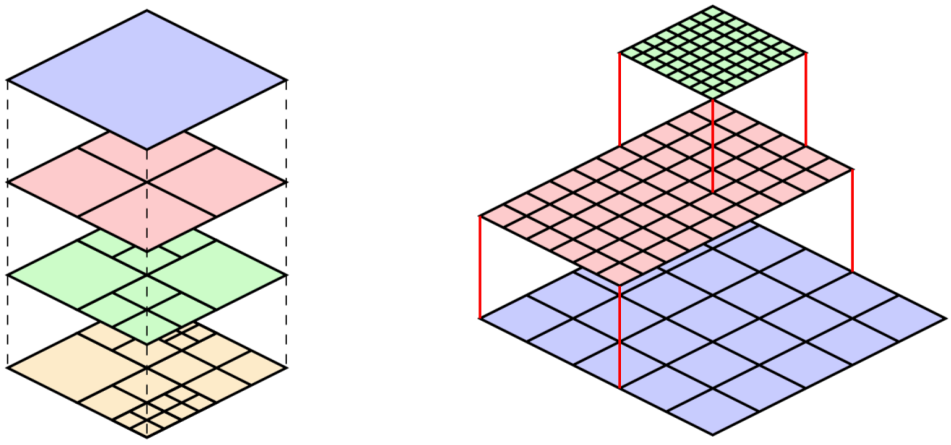

Рисунок 1 Представление сетки: cell-based подход (слева), block-based подход (справа)

2.3.2 Блочно-ориентированный подход

Блочно-ориентированный (block-based) подход основан на представлении сетки в виде нескольких слоев, каждый из которых представляет собой набор блоков ячеек. Если блоки составляют прямоугольное покрытие области измельчения, то подход называется патчевым (patch-based). Если блоки имеют одинаковый размер, то данный подход называется плиточным (tile-based). Блоки одного уровня обычно объединяются в некоторую структуру, и далее полученные структуры хранятся и балансируются независимо. Схематичное изображение данного подхода представлено на ***Рисунок 2***.

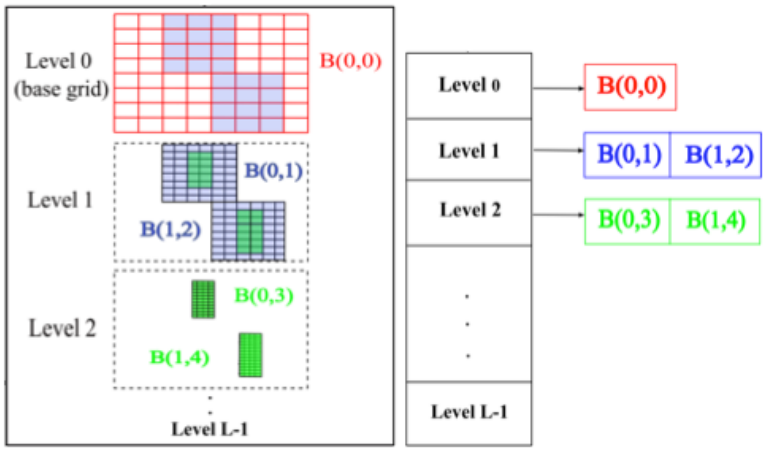

Рисунок 2 Представление расчетной сетки при блочно-ориентированном подходе

Базовые концепции данного подхода описаны в [18]. Для блочно-ориентированного подхода определяется последовательность уровней $l = 1, \dots, l_{max}$. Сетка $G_{l,k}$ имеет пространственный шаг h_l и номер k на уровне l и называется блоком. Уровень 1 имеет наибольший пространственный шаг. Сетка $G_l = \bigcup_k G_{l,k}$, называемая сеткой уровня или уровнем. Расчетная область является первым уровнем, то есть $G_1 = \bigcup_k G_{1,k}$. Сетки уровней в должны быть правильно вложенными (proper nested). Это означает, что

1. каждая ячейка блоков уровня l должна лежать внутри некоторого блока уровня $l - 1$, и при этом количество ячеек вдоль одной границы блока уровня l должно быть кратно некоторому числу, называемому коэффициент измельчения (refinement factor), обычно равному 2;
2. для каждой ячейки уровня l в каждом направлении (север, юг, восток, запад), кроме направления к границе области, должна быть хотя бы одна ячейка на уровне $l - 1$.

Построение сетки происходит итеративно для каждого уровня и состоит из трех этапов: оценка ошибки (градиента) и пометка ячеек для измельчения, покрытие помеченных ячеек

прямоугольниками, создание блоков нового уровня. Условие, при котором ячейка помечается к измельчению, называется критерием измельчения. Анализ на основе данного критерия аналогичен для любого способа представления декартовой локально-сгущающейся сетки. Для покрытия областей помеченных ячеек прямоугольниками (clustering) используются алгоритмы из анализа бинарных изображений и компьютерного зрения [19]. При покрытии важно, чтобы сохранялось условие правильной вложенности. При создании блоков следующего уровня формируется набор блоков измельченных ячеек, соответствующих прямоугольникам, полученных на предыдущем шаге. Также обычно в блоки по границам добавляются несколько рядов буферных ячеек. Отсюда следуют два ключевых момента: во-первых, расчетная сетка может иметь только прямоугольную форму, и, во-вторых, форма областей измельчения также должна быть достаточно простой для получения качественного покрытия ее прямоугольниками.

Стоит заметить, что временной шаг определяется пространственным шагом: чем меньше размер ячейки, тем меньше должен быть пространственный шаг. При использовании локально-сгущающихся сеток, организованных блочно, есть возможность реализовать в каждом блоке временной шаг, соответствующий размеру ячеек блока. Для этого используются подциклы (sublooping): на один временной шаг для ячеек базового, первого, уровня делается несколько временных шагов следующих уровней. Количество дополнительных шагов зависит от уровня, а от количества дополнительных шагов зависит количество необходимых рядов буферных ячеек.

Основными программными пакетами, реализующими блочно-ориентированный подход, являются Chombo, Enzo, Uintah. В статье [21] проводится обзор и сравнение этих программных пакетов. Отличия состоят в первую очередь в алгоритмах покрытия измельчаемых областей блоками (LBR, GBR, Tiled) и подходах к организации связей между уровнями и соседями. Все эти библиотеки эффективно организуют обращение к памяти и использование регулярных структур данных для работы с блоками.

2.3.3 Ячейко-ориентированный подход

Ячейко-ориентированнй (cell-based) подход основан на независимом разбиении ячеек. При данном подходе структура сетки в целом образует четверичное (восьмеричное в трехмерном случае) дерево.

Сухинов А.А. в [13] описывает древовидную структуру представления ячейко-ориентированных локально-сгущающихся декартовых сеток.

Каждая ячейка C_i имеет размеры (h_i^x, h_i^y) и хранит величину V_i (скалярную или векторную), описывающую значения вычисляемой функции в пределах ячейки — значение ячейки.

Ячейка может быть разбита на четыре ячейки одинаковых размеров – дочерние ячейки (потомки). Разделенная ячейка называется родителем своих потомков. При этом ячейка-родитель не удаляется. Четыре ячейки могут быть объединены только если они являются прямыми потомками одной ячейки. Объединение одной ячейки — удаление ее потомков.

Окрестность ячейки C_i — прямоугольник размером $(3h_i^x, 3h_i^y)$ с центром в центре самой ячейки C_i. Соседи ячейки C_i — восемь (возможно совпадающих) ячеек окрестности $C_{nLU(i)}$, $C_{nU(i)}$, $C_{nRU(i)}$, $C_{nL(i)}$, $C_{nR(i)}$, $C_{nLB(i)}$, $C_{nB(i)}$, $C_{nRB(i)}$ (**Рисунок *3***). При этом считаем, что соседние ячейки могут отличаться по размеру не более чем в два раза для того, чтобы сетка удовлетворяла критерию гладкости. Соседи ячейки могут иметь потомков, могут быть больше по размеру, чем сама ячейка. Также можно найти двенадцать малых соседей ячейки — потомков основных соседей.

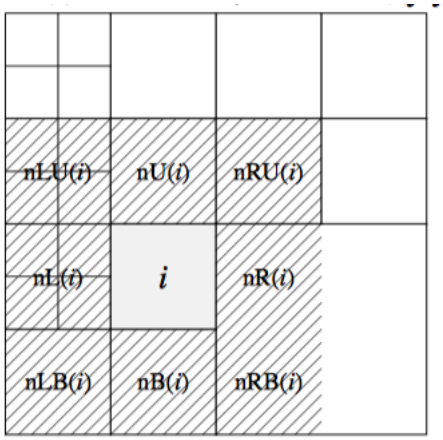

Рисунок 3 Окрестность ячейки

При построении сетки исходно считаем, что она состоит из одной ячейки C_0 — корневой ячейки сетки. Далее путем последовательных разбиений ячеек получим четверичное дерево. Вычисления задачи производятся в его листьях (**Рисунок *4***). Для хранения древовидной структуры необходимо в ячейке C_i хранить индексы дочерних ячеек (указатели на них) $C_{cLU(i)}$, $C_{cLB(i)}$, $C_{cRU(i)}$, $C_{cRB(i)}$ и индекс родительской ячейки (указатель на нее) $C_{P(i)}$.

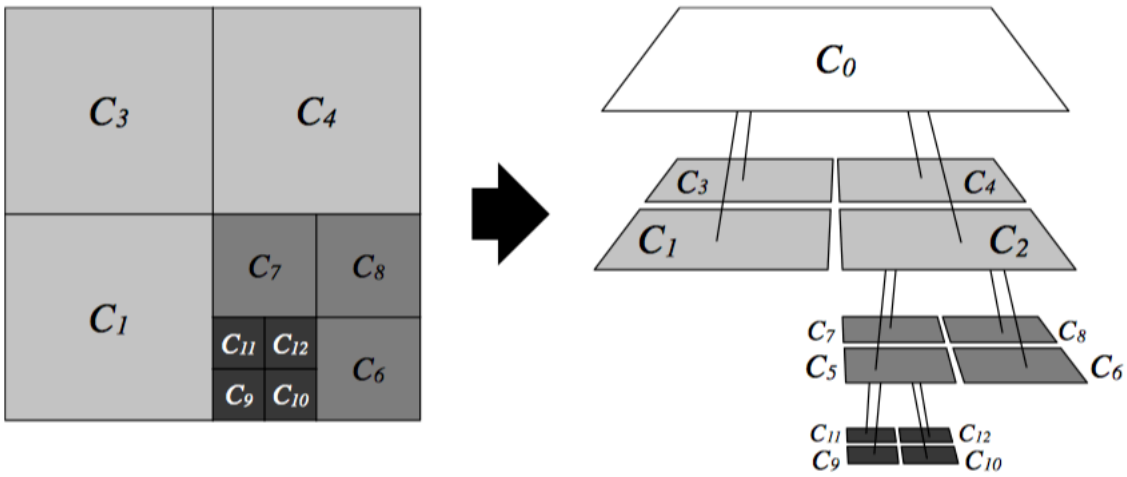

Рисунок 4 Дерево, представляющее двумерную декартову локально-сгущающуюся сетку

Определим адрес ячейки C_i как путь от корневой ячейки C_0 до ячейки C_i, записанный в виде двух двоичных строк (X_i, Y_i) следующим образом:

1. Корневая ячейка имеет адрес, состоящий из двух пустых строк: $(X_0, Y_0) = (\{\}, \{\})$.
2. Адрес ячейки C_i может быть получен из адреса ее родителя по формуле $(X_i, Y_i) = \left(X_{P(i)} \oplus x_i, Y_{P(i)} \oplus y_i\right)$, где $\oplus$ — оператор конкатенации, $x_i = 0$, если C_i — один из левых потомков $C_{P(i)}$, и $x_i = 1$ иначе; $y_i = 0$, если C_i — один из нижних потомков $C_{P(i)}$, и $y_i = 1$ иначе.

Глубина ячейки d_i — число разрядов в компонентах адреса. $d_i = |X_i| = |Y_i|$.

Адрес ячейки однозначно задает набор из трех целых чисел со знаком: глубина ячейки d_iв дереве и координаты на данном уровне (X_i, Y_i). При таком способе хранения максимальная глубина ячейки на 32-хбитной системе составит 32 уровня, то есть до 2^{32} листовых ячеек. Два старших бита компонент можно зарезервировать для отслеживания граничных условий, тогда максимальная глубина составит 30 уровней и до 2^{30} листовых ячеек.

Параметры ячеек могут быть вычислены следующим образом. Размер ячейки: $\left(h_i^x, h_i^y\right) = \left(\frac{h_0^x}{2^{d_i}}, \frac{h_0^y}{2^{d_i}}\right)$. Координаты левого нижнего угла ячейки: $x_i = h_0^x \cdot \frac{X_i}{2^{d_i}}$, $y_i = h_0^y \cdot \frac{y_i}{2^{d_i}}$.

Адреса соседей ячейки могут быть найдены с помощью двоичного сложения и вычитания единицы из компонент адреса ячейки: $C_{nLU(i)}\,(X_i - 1, Y_i + 1)$; $C_{nU(i)}: (X_i, Y_i + 1)$; $C_{nRU(i)}: (X_i + 1, Y_i + 1)$; $C_{nL(i)}: (X_i - 1, Y_i)$; $C_{nR(i)}: (X_i + 1, Y_i)$; $C_{nLB(i)}: (X_i - 1, Y_i - 1)$; $C_{nB(i)}: (X_i, Y_i - 1)$; $C_{nRB(i)}: (X_i + 1, Y_i - 1)$, где $+, -$ — операции двоичного сложения и вычитания. Битовая единица может быть представлена числом 2^{30-d_i}.

Основными программными пакетами, реализующими ячейко-ориентированный подход, являются RAMSES [27], p4est [20]. Отличия состоят в способе представления дерева ячеек: первый реализует подход к представлению, описанный выше, в виде полного дерева с одним дополнением: кроме указателей на потомков и родителя ячейки сохраняются также указатели на соседей на том же уровне; второй реализует подход компактного представления дерева ячеек в виде линейного массива. Также возможно использование хэш-таблицы для представления дерева ячеек [2].

Основная проблема структур полного дерева и хэш-таблицы — нерегулярный доступ к памяти. К тому же при использовании полного дерева для хранения сетки, используется избыточное количество памяти для хранения нелистовых ячеек. Представление в виде линейного массива дает наибольшую эффективность как по объему используемой памяти, так и по эффективности доступа к ней.

Для построения линейного массива для хранения сетки необходимо построить биективное отображение $(d_i, X_i, Y_i) \leftrightarrow m$, где m — глобальный индекс ячейки. В качестве данного отображения обычно используется кривая Мортона, или z-кривая (**Рисунок *5***), задающееся следующим образом (**Рисунок *2***). Если $x_0x_1\ldots x_{31}$ — двоичная запись X_i, а $y_0y_1\ldots y_{31}$ — двоичная запись числа Y_i, то $m_0m_1\ldots m_{63}$, где $m_{2k} = x_k$, $m_{2k+1} = y_k$ — двоичная запись глобального индекса m ячейки C_i. В линейный массив записываются ячейки в порядке полученных глобальных индексов.

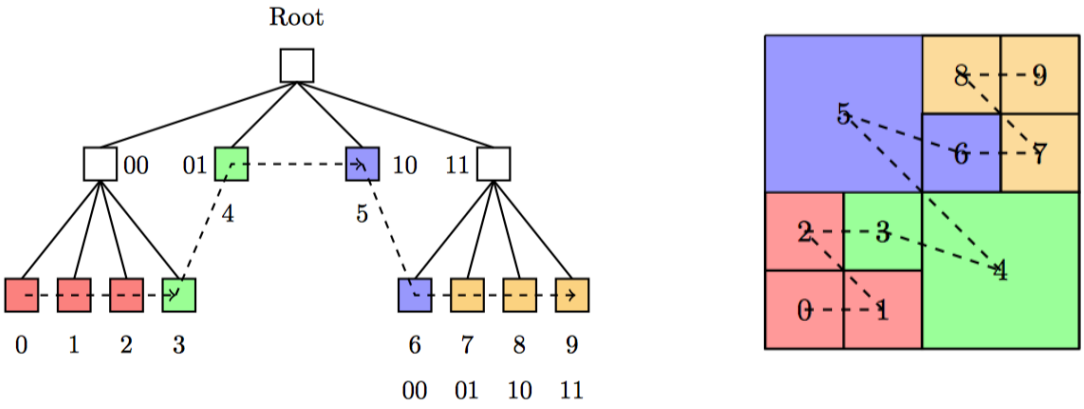

Рисунок 5 Обход четверичного дерева в порядке кривой Мортона

Кривая Мортона выбрана среди различных кривых, заполняющих пространство, потому что, во-первых, обеспечивает сохранение древовидной структуры, таким образом упрощая адаптацию, во-вторых, сохраняет индексирование, таким образом обеспечивая однозначность координат, в-третьих, имеет наибольшую локальность доступа к памяти и эффективность кэша [25].

Заметим, что при компактном хранении дерева в виде линейного массива последовательная индексация в массиве ячеек не соответствует индексам по кривой Мортона, а значит необходимо сохранять в каждой ячейке указатели на соседей. Для поиска соседа необходимо с помощью описанных выше битовых операций получить номер ячейки-соседа, а затем найти ее в отсортированном в порядке глобальных индексов массиве ячеек с помощью бинарного поиска.

Таким образом, при использовании ячейко-ориентированного подхода можно хранить в памяти только расчетные ячейки, а значит объем памяти, необходимый для представления расчетной сетки, меньше, чем при использовании блочно-ориентированного подхода.

Также заметим, что выше при построении дерева предполагается что существует одна корневая ячейка, однако можно обобщить алгоритмы на случай с несколькими корневыми ячейками и получить представление расчетной сетки в виде леса деревьев.

2.3.4 Процесс адаптации

Адаптация сетки включает в себя две операции: измельчение — переход от более грубой сетки к более мелкой, и огрубление — переход от измельченной сетки к более грубой.

Адаптация обычно происходит в три этапа:

1. Процедура анализа сетки: производится анализ ячеек на основе критерия адаптации, помечаются ячейки, для которых выполняется критерий измельчения или выполняется критерий огрубления и огрубление возможно. Процедура проводится для всех уровней, начиная с первого.
2. Процедура измельчения/огрубления сетки: помеченные ячейки измельчаются/огрубляются. Процедура выполняется для уровней от нулевого до предпоследнего.
3. Процедура сглаживания сетки (2:1 balancing): ячейки, у имеющие соседей мельче более чем в два раза, измельчаются до нужного уровня. Процедура выполняется для уровней с первого до предпоследнего.

Анализ сетки происходит локально (на вход анализатора подаются данные о ячейке и ее окрестности) согласно некоторому критерию адаптации, зависящему от типа задачи. Если решение известно, то подходящим критерием может быть $\left\| \nabla f \right\| h > \varepsilon$, где f — решение, h f — размер ячейки, ε — некоторый порог адаптации. Существуют алгоритмы алаптации, основанные на алгоритмах области анализа, например детектор границ Кэнни, определяющий локальные максимумы градиента в направлении его вектора. Для более точного анализа могут использоваться критерии измельчения на основе вейвлетов, данный

подход активно исследуется, например, в [3]. С алгоритмической точки зрения пометка ячеек сопоставима с одним временным шагом, так как включает в себя обход всей сетки и анализ значений соседей.

Алгоритм измельчения сетки зависит от способа представления: в случае блочного подхода — это определение покрытия областей помеченных ячеек блоками и создание новых блоков; в случае ячейко-ориентированного подхода с использованием полного дерева — это создание новых элементов-ячеек; в случае ячейко-ориентированного подхода с использованием линейного массива — это вставка и удаление некоторых элементов массива. Для сокращения количества модификаций массива обычно измельчаются не ячейки по отдельности, а отрезки последовательных помеченных ячеек, а также в качестве промежуточной структуры данных используется список. При измельчении необходимо производить перенос значений с более крупной сетки на более мелкую, для этого можно использовать различные методы, анализ и сравнение данных методов также проводится в [3].

Сглаживание сетки аналогично совокупности первых двух процедур, где анализ производится на основе значений уровней соседних ячеек.

2.3.5 Метод конечных объемов

Для решения дифференциальных уравнений в частных производных существует три группы методов: метод конечных разностей (МКР), метод конечных элементов (МКЭ) и метод конечных объемов (МКО). Для интегрирования дифференциальных уравнений на декартовых локально-измельченных сетках наиболее распространен метод конечных объемов, основанный на интегральной формулировке законов сохранения.

Решение дифференциального уравнения в частных производных с использованием метода МКО происходит следующим образом. Область разбивается на небольшие соприкасающиеся объемы. Для каждого такого объема записывается балансовое соотношение: $\int_{\Omega} \frac{\partial \rho \phi}{\partial t} d\Omega + \sum_{k} \int_{S_k} \vec{n} \bullet \vec{q}\, ds = \int_{\Omega} Q d\Omega$, $\vec{q} = \rho \vec{V} \phi - \alpha \nabla \phi$, где $\vec{q}$ — вектор плотности потока величины ϕ, включает в себя диффузионную и конвективную составляющие; Q — плотность распределения объемных источников; $\vec{V}$ — вектор скорости, ρ — плотность среды; α — коэффициент диффузии. При стремлении объема к нулю, получаем дифференциальную форму закона сохранения: $\frac{\partial \rho \phi}{\partial t} + \nabla \bullet \vec{q} = Q$.

Внутри каждого объема выбирается одна точка привязки искомого сеточного решения. Обычно в качестве объема выбирается ячейка сетки, в качестве точки привязки — ее центр.

Для получения дискретного аналога необходимо вычислить интегралы балансового соотношения с помощью квадратурных формул. Для того, чтобы схема была консервативна, необходимо чтобы интегралы соседних объемов по общей грани вычислялись идентично. Свойства численной схемы (устойчивость, точность, монотонность) зависят от выбора способа интегрирования.

2.4 Методы декомпозиции расчетных сеток

При численном решении физических задач с использованием суперкомпьютеров широко используется геометрический параллелизм, при котором расчетная сетка разбивается на домены, обрабатываемые отдельными процессорами. Основная проблема при его использовании — необходимость решения задачи статической балансировки. Статическая балансировка нагрузки — выбор такой декомпозиции, при котором вычислительная нагрузка распределена равномерно между процессорами, а накладные расходы, вызванные дублированием вычислений и необходимостью передачи данный, малы [15]. При исследовании данного вопроса расчетную сетку обычно рассматривают как граф, вершинами (узлами) которого являются ячейки расчетной сетки, а ребра обозначают соседство ячеек. Вершинам и ребрам могут быть присвоены веса, соответствующие объему вычислений в ячейке и объему передачи данных между соседними ячейками соответственно. Основными критериями декомпозиции графов являются равномерное распределение по доменам суммарного веса узлов/ребер и минимизация максимального веса исходящих из домена ребер. Одновременное выполнение данных требований является сложной задачей, поэтому обычно выбирается некоторый баланс между. Обзор методов декомпозиции приводится в [15].

2.4.1 Декомпозиция на основе нумерации узлов

Самым простым методом декомпозиции является декомпозиция на основе исходной нумерации узлов. Для более качественной декомпозиции данным методом применяют перенумерацию узлов в порядке кривых, заполняющих пространство.

Обычно расчетная сетка представляется в виде множества узлов с некоторой заданной нумерацией. Тогда для разбиения сетки из n узлов на p доменов можно применить следующий алгоритм: отнесем узлы с номерами от $k\frac{n}{p}$ до $(k+1)\frac{n}{p}-1$ к домену с номером k.

Методы декомпозиции на основе нумерации узлов просты в реализации и позволяют получить разбиение с хорошим балансом нагрузки за небольшое время, однако он никак не учитывает связи между узлами, что может приводить к тому, что границы между доменами

проходят по большому числу ребер и к соседству с большим числом доменов, и, следовательно, к увеличению коммуникационной нагрузки.

2.4.2 Методы рекурсивной бисекции

При использовании метода рекурсивной бисекции сетка за $k-1$ шаг разбивается на k частей, на каждом шаге одна из частей, полученных на предыдущем шаге, разбивается на две новых (бисекция). Этот процесс можно представить в виде бинарного дерева.

Процедура непосредственно бисекции может быть выбрана в зависимости от вида расчетной сетки. Например, для регулярных сеток можно использовать индексную бисекцию, которая имеет преимущество по скорости, однако на выходе дает домены сложной формы, что стирает преимущество регулярности расчетной сетки. Для произвольных графов можно использовать координатную бисекцию с последовательным выбором направления разреза и сортировкой вершин в направлении выбранной оси при разрезании.

2.4.3 Графовый подход

В общем случае задача оптимальной декомпозиции является аналогом задачи о разрезании графа, то есть является NP-полной. Алгоритмы нахождения точного решения задачи декомпозиции имеют экспоненциальную сложность, а значит неприменимы в реальных задачах. Поэтому обычно используются эвристические алгоритмы. Основным используемым принципом является иерархический подход.

Иерархическая декомпозиция представляет из себя алгоритм, состоящий из четырех основных шагов: огрубление графа, декомпозиция огрубленного графа, восстановление графа, локальное уточнение границ. Огрубление графа — построение последовательности уменьшающихся в размере вложенных графов, основано на последовательном стягивании ребер, таких что никакая из вершин сетки не инцидентна более чем одному из выбранных ребер. Декомпозиция огрубленного графа — распределение вершин огрубленных графов по заданному числу доменов, для этого могут использоваться различные алгоритмы декомпозиции, такие как координатная бисекция, спектральная бисекция (основана на спектральной матрице графа, минимизирует суммарный вес разрезаемых ребер). Восстановление графа — процедура обратная огрублению, при этом восстановленные вершины включаются в тот же домен, что и вершины, их порождающие. Локальное уточнение границ доменов — выравнивание весов доменов исходного графа, позволяет улучшить баланс нагрузки и уменьшить вес разрезаемых ребер.

Существует также другой подход к декомпозиции графов, не использующий огрубление, ориентированный на получение связных и компактных доменов — алгоритм

инкрементного роста. Данный подход основан на выделении ядер графа и итерационном применении процедур: инициализация доменов — определение случайно выбранных вершин в количестве необходимого количества доменов, инкрементный рост — распределение вершин по доменам, локальное уточнение границ, анализ связности ядер полученных доменов, перенос части закрепленных за доменами вершин.

Для декомпозиции ячейко-ориентированных локально-сгущающихся расчетных сеток обычно используется метод декомпозиции на основе нумерации узлов, которая строится на основе кривой Мортона. Данный способ позволяет быстро получить качественное разбиение с точки зрения баланса нагрузки, однако границы полученных доменов криволинейны и не оптимальны по длине, а также домены могут получаться разрывными из-за скачков кривой. Также заметим, что для блочно-ориентированных декартовых сеток декомпозиция является более сложной задачей из-за неплоской структуры сетки и межуровневых связей. Использование для них классических алгоритмов декомпозиции ведет к необходимости трудоемкому переводу сетки в графовое представление. Особенно усложняется задача, если блоки более высоких уровней могут покрывать части нескольких блоков предыдущих уровней.

2.5 Алгоритмы обработки доменов

При использовании суперкомпьютеров возникает необходимость разработки параллельных алгоритмов обработки полученных на этапе декомпозиции доменов расчетных сеток на общей и распределенной памяти. При использовании гибридного параллелизма в виду иерархической организации памяти вычислительных узлов, существенную роль играет локальность доступа к памяти. Если принцип локальности нарушается, увеличивается количество промахов кэша, и, следовательно, падает производительность. В [4] проводится анализ эффективности распараллеливания вычислений на неструктурированных сетках, однако описанные принципы можно применить и к декартовым локально-сгущающимся сеткам. Основным принципом является использование двухуровневой модели сеток: сначала сетка разбивается на микродомены, размер которых оптимален для использования кэш-памяти конкретной вычислительной машины, а затем обработка сетки производится для каждого из этих микродоменов последовательно. Единицей балансировки в данном случае также является микродомен. Были проведены исследования, показывающие, что использование такого подхода повышает эффективность выполнения расчетов на таких сетках за счет большей локальности обращений к памяти.

Также важной проблемой при использовании гибридного параллелизма является пересечение по данным между нитями, то есть наличие критических секций и атомарных

операций. Для того чтобы избежать при этом падения эффективности можно, во-первых, продублировать данные, необходимые для нескольких нитей, во-вторых, применить следующий подход: сначала обработать непересекающиеся области, а затем интерфейсные элементы. Первый вариант ведет к возрастанию необходимой памяти, то есть снижает эффективность. Второй почти не снижает эффективности вычислений так как количество интерфейсных элементов обычно значительно меньше, чем внутренних, а объем вычислений не возрастает.

2.6 Выводы

Из проведенного обзора были сделаны следующие выводы:

1. Генерация качественных неструктурированных сеток — трудоемкая и не до конца решенная задача. Существующие программы-генераторы не удовлетворяют требованиям множественного перестроения сетки для решения задач аэродинамического обтекания. В то же время генерация декартовых сеток, покрывающих область, проста в реализации. Для того чтобы декартова локально-измельченная сетка удовлетворяла критерию гладкости необходимо чтобы соседние ячейки отличались не более чем в два раза.
2. Ячейко-ориентированные локально-сгущающиеся декартовы сетки эффективны по объему памяти, дают гибкие возможности для декомпозиции.
3. Основной вычислительной нагрузкой при использовании ячейко-ориентированных сеток является поиск и перестроение структур соседей, чтобы минимизировать время на поиск соседей, необходимо хранить указатели на соседей, что влечет за собой существенные накладные расходы памяти, и, следовательно, существенно снижает эффективность использования кэша.
4. Наиболее естественным и эффективным для ячейко-ориентированных локально-сгущающихся декартовых сеток является метод разбиения по кривой Мортона. Однако при декомпозиции ячейко-ориентированной сетки по кривой Мортона, границы доменов имеют сложную для обработки форму, и, следовательно, увеличивают время упаковки данных для обменов между вычислительными узлами. При использовании блочного подхода для использования классических алгоритмов декомпозиции необходимо чтобы блоки более высокого уровня покрывали только часть одного блока более низкого уровня.
5. Для эффективного использования гибридного параллелизма необходимо, чтобы шаблон доступа к памяти был регулярным и локальным. Для этого удобно разбить

сетку на микродомены, соответствующие объемы кэш-памяти машины. При этом удобно на общей памяти параллельно обрабатывать непересекающиеся области этих микродоменов, а затем их границы.

3. Постановка задачи

Необходимо предложить структуру данных для представления декартовых локально-сгущающихся сеток, улучшающую свойства ячейко-ориентированного подхода, то есть такую, чтобы шаблон доступа к памяти был регулярным, и такую, чтобы накладные расходы на обработку соседства были меньше. Необходимо разработать алгоритм построения сетки с предложенной структурой, адаптирующейся к начальным условиям.

Для структуры сетки необходимо разработать эффективный алгоритм статической декомпозиции на домены, то есть отображения множества ячеек сетки на множество процессоров.

Для предложенной структуры сетки и домена необходимо разработать алгоритмы эффективной гибридной параллельной обработки полученных доменов в предположении использования явных конечно-объемных методов расчета.

4. Предложенное решение

В данном разделе описана структура данных для представления декартовых локально-сгущающихся сеток, алгоритм ее построения, алгоритм декомпозиции расчетной сетки на домены в данном представлении и алгоритм обработки одного домена вычислительным узлом.

4.1 Представление сетки

Локально-сгущающиеся декартовы сетки удобно и эффективно представлять в памяти в виде линейного массива с ячейками в порядке кривой Мортона. Однако для более эффективного поиска соседей и более локализованного обращения к памяти необходимо выделить блоки ячеек одного уровня. Однако выделение блоков в понимании блочно-ориентированного подхода приводит к стиранию структуры сетки. Поэтому для организации структуры сетки предлагается использовать дерево как в ячейко-ориентированном подходе. Данное дерево можно так же компактно представить в виде линейного массива, содержащего блоки ячеек, отсортированные в порядке кривой Мортона. При этом каждый блок содержит в себе ячейки одного уровня в виде обычной матрицы, хранящейся в памяти в виде одного блока для повышения локальности доступа. Схематически предложенное представление изображено на ***Рисунок 6***.

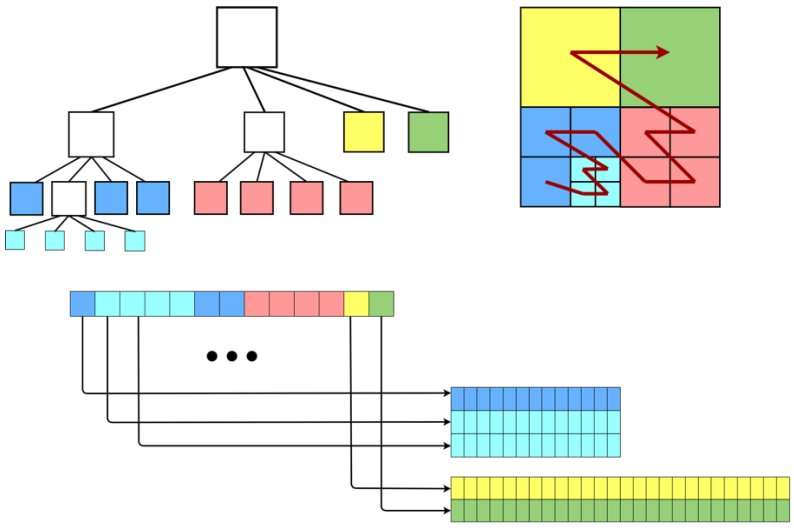

Рисунок 6 Компактное представление расчетной сетки в виде дерева блоков

Таким образом мы сохраняем структуру сетки и получаем «плоскую» структуру для блоков разных уровней. При этом при обработке блока адреса соседей ячеек внутри блока известны и близки к ней, алгоритмически сложный поиск соседей необходимо выполнять лишь для

блоков, которых много меньше, чем ячеек. Более того, для блоков ограничение на различие размеров соседних ячеек не более чем в два раза не распространяется, хотя и сохраняется для самих ячеек.

Уровни базовой сетки и блока можно задать числовыми параметрами ($base\_lvl$, $base\_blk\_lvl$), глубину дерева блоков можно варьировать с помощью числового параметра максимального уровня блока (max_blk_lvl), а общее количество ячеек с помощью максимального уровня сетки (max_lvl).

Заметим, что в таком представлении для нумерации непосредственно ячеек можно использовать тот же подход, что для ячейке-ориентированных сеток. Глобальный номер ячейки будет состоять из двух частей: номера блока и номера внутри блока, и может быть вычислен с помощью битовых операций. Таким образом все алгоритмы, определенные для ячеко-ориентированных сеток, могут быть применены к сеткам, представленным в предложенном виде.

Алгоритм построения дерева блоков происходит по аналогии с построением дерева ячеек следующим образом (***Рисунок 7***):

1. Покрытие области блоками базового уровня блоков, содержащими ячейки базового уровня ячеек.
2. Анализ ячеек каждого блока с использованием критерия измельчения, пометка ячеек и блоков. Блок помечается к измельчению, если он содержит помеченные ячейки.
3. Помеченные блоки разбиваются на четыре подблока. Подблок помечается к измельчению если он содержит помеченные ячейки.
4. Ячейки помеченных блоков измельчаются.

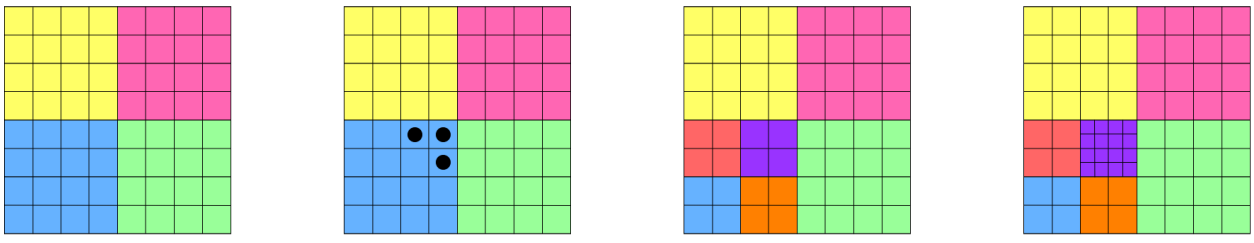

Рисунок 7 Построение дерева блоков: базовое покрытие, пометка ячеек, разделение блоков, измельчение ячеек

Для построения структуры соседей предлагается следующий алгоритм:

1. Для каждого блока формируем списки возможных соседей того же уровня. Количество возможных соседей всех уровней по одной грани определяется по

параметрам минимального и максимального возможных уровней блоков по формуле $2^{max\_blk\_lvl-base\_blk\_lvl}$.

2. Для каждого элемента списка возможных соседей целевого блока осуществляем бинарный поиск в массиве блоков. Возможны три варианта:
    a. элемент с данным индексом найден и не найдены элементы с индексами, имеющими такой же уровневый префикс – тогда данный элемент является реальным соседом целевого блока, записываем указатель на него в массив соседей целевого блока;
    b. элемент с данным индексом найден и, кроме того, найдены элементы с индексами, имеющими такой же уровневый префикс – тогда реальными соседями целевого блока являются все ближайшие к целевому блоку потомки данного элемента, записываем указатели на них в массив соседей целевого блока;
    c. элемент с данным индексом не найден – тогда реальным соседом является некоторый родитель данного элемента, пропускаем данный элемент на этом этапе.
3. Таким образом мы получили списки реальных соседей не большего уровня чем целевые блоки.
4. Проходим по всем блокам по массивам реальных соседей и добавляем указатель на себя в массив соседей соседа, если его там еще нет.

Предложенная структура сочетает в себе эффективность по объему памяти и гибкость декомпозиции ячейко-ориентированных сеток, и возможность использования регулярного шаблона доступа к памяти блочно-ориентированного подхода. Заметим, что количество ячеек, а значит объем вычислений, может увеличиться по сравнению с ячейко-ориентированным подходом.

4.2 Алгоритм декомпозиции

В данной работе предлагается использовать алгоритм декомпозиции по кривой Мортона на уровне блоков.

Определим вес блока как количество ячеек в нем. Определим домен как набор последовательных блоков, принадлежащих одному вычислительному узлу. Определим вес домена как сумму весов входящих в него блоков. Определим оптимальный вес как вес, такой что веса всех доменов равны. Оптимальный вес для разбиения на p доменов можно

вычислить по формуле $\frac{\sum_i w_i}{p}$, где в числителе стоит сумма весов всех блоков расчетной сетки, а в знаменателе —- количество доменов, на которые разбивается сетка. Тогда можем производить декомпозицию с учетом вычислительной нагрузки следующим образом:

1. вычисляем суммарный вес всех блоков;
2. вычисляем оптимальный вес домена;
3. проходим по массиву блоков от первого до последнего с накоплением весов;
4. когда текущий вес начинает превышать оптимальный, считаем, что начинается отсчет следующего домена.

Преимуществами декомпозиции массива блоков перед декомпозицией массива ячеек является то, что границы доменов получаются более ровными, более того, при обменах мы заранее знаем, какие ячейки граничные, а какие внутренние, что упрощает заполнение значений в граничных ячейках и упаковку данных для обменов.

Домен расчетной сетки в предложенном представлении — это массив последовательных блоков. Однако для организации параллельной обработки необходимо на каждом процессоре хранить некоторую метаинформацию, позволяющую определить связи с другими процессорами. Для организации обменов необходимо иметь информацию о диапазонах номеров блоков на каждом процессоре, чтобы определить, кому принадлежит необходимые блоки и ячейки, а также информацию о блоках, соседствующих с блоками домена: уровни блока и ячеек, глобальный номер для определения координат. Также необходимы буферы обмена фантомными ячейками. В случае статической балансировки нагрузки инициализацию данных структур можно производить распределенно один раз при инициализации расчетной сетки. Для этого необходимо:

1. Построить фантомные блоки — блоки, соседствующие с блоками данного домена, и обрабатываемые на других процессорах. Фантомные блоки не содержат ячеек, а содержат лишь информацию об уровне и размере блока, необходимую для “сшивания” границ блоков. Для построения структуры фантомных блоков необходимо несколько обменов типа точка-точка с соседними процессорами.
2. Построить фантомные ячейки —- соседние граничные ячейки фантомных блоков. Фантомные ячейки должны содержать глобальный индекс и значение целевой функции для текущего временного слоя. Для определения индексов фантомных ячеек и для их поиска можно воспользоваться описанным выше сходством с ячейке-

ориентированными сетками. Для построения структуры фантомных ячеек необходимо несколько обменов типа точка-точка с соседними процессорами.

4.3 Алгоритм обработки домена

Для эффективного использования общей памяти необходимо сначала обрабатывать непересекающиеся области сетки, а затем интерфейсные элементы. Этот подход удобно реализовать для предложенного представления расчетной сетки. Также стоит учесть, что можно сократить активное время коммуникаций с соседями, если использовать асинхронные операции типа точка-точка. Итак, предлагается следующий алгоритм для обработки домена вычислительным узлом (для одного временного шага):

1. Инициализация обмена фантомами ячейками: неблокирующие отправка и прием.
2. Обработка внутренних ячеек блоков: обработка блоков равномерно распределяются по потокам.
3. Ожидание завершения обмена фантомными ячейками.
4. Обработка граничных ячеек блоков: обработка блоков происходит в одном потоке с обращениями к ячейкам других блоков и фантомом ячейкам.

Преимущество предложенного алгоритма состоит в использовании регулярного и локального шаблон доступа к памяти внутри блоков.

4.4 Возможные дальнейшие оптимизации

Предложенное решение можно оптимизировать исходя из проведенного обзора.

Во-первых, вообще говоря, если мы обладаем достаточным диапазоном уровней блоков, то мы можем провести более глубокий анализ необходимости измельчения, и сократить количество измельчаемых ячеек.

Во-вторых, стоит заметить, что в данном представлении сетки возможно использовать как классическое разбиение по кривой Мортона, так и графовые алгоритмы балансировки, в которых узлом будет являться блок, имеющий вес – количество ячеек, которое он содержит. Для использования графовых алгоритмов необходима процедура приведения сетки к графовому виду, аналогичная представленной ниже процедуре поиска соседей, а также алгоритмы работы с массивами кусочно-последовательных блоков, которые выходят за рамки данной работы.

5. Экспериментальное исследование

Для исследования эффективности различных подходов к распределению нагрузки были реализованы программы на языке C++ с использованием технологий MPI и OpenMP, соответствующие предложенному подходу и подходу p4est, для построения, декомпозиции декартовых локально-сгущающихся расчетных сеток и численного решения модельной задачи на их основе.

5.1 Модельная задача

Предлагается рассмотреть модельную задачу распространения тепла в двумерной области с точечным источником тепла и заданным начальным распределением, под которые необходимо адаптировать сетку.

В области $D = D_{xy} \times [0 < t < T], D_{xy} = [0,1] \times [0,1]$ ищется решение двумерного уравнения теплопроводности

$$\frac{\partial u}{\partial t} = \alpha \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) + q(x, y, t), \ \alpha = 0.01,$$

где q —- функция источника, задается формулой

$$q(x, y, t) = \begin{cases} 0.01, & (x - c_x)^2 + (y - c_y)^2 < r \\ 0, & (x - c_x)^2 + (y - c_y)^2 \geq r \end{cases}, \quad c_x = c_y = 0.25, \ \ r = 0.01,$$

удовлетворяющее граничным условиям

$$\frac{\partial u(0,y,t)}{\partial y} = \frac{\partial u(1,y,t)}{\partial y} = \frac{\partial u(x,0,t)}{\partial x} = \frac{\partial u(x,1,t)}{\partial x} = 0$$

и начальному условию

$$u(x, y,0) = \frac{1}{\sigma_x \sigma_y} exp\left(-\frac{(x - m_x)^2}{\sigma_x^2} - \frac{(y - m_y)^2}{\sigma_y^2} \right), \ \ m_x = m_y = 0.25, \ \ \sigma_x = \sigma_y = 0.05.$$

Численная аппроксимация для температуры в элементе объема производится следующим образом [2]:

$$u_i^{n+1} = u_i^n - \ \tau \ \frac{\left(\sum_{k-соседн.} s_{ik} \left(F_{ik}^x + F_{ik}^y \right) \right) - \frac{q_i}{c\rho}}{S_i}, \ \ F_{ik}^x = \ F_{nk} \cos(\hat{F_{nk}, O_x}) \ , \ F_{ik} = \ \ - \alpha \frac{u_k^n - u_i^n}{dist(i, \ k)},$$

где S_i— площадь ячейки i, s_{ik}— длина грани между ячейками i и k, F_{ik} — поток тепла между ячейками i и k, F_{ik}^x и F_{ik}^y — проекции потока тепла по направлениям, параллельным осям координат.

5.2 Построенные сетки

Все эксперименты проводились с использованием декартовой локально-измельченной расчетной сетки с четырьмя уровнями измельчения. В ***Таблица 1*** приведены сравнительные характеристики расчетных сеток, построенных для ячейко-ориентированного подхода и для предложенного решения, использующиеся для исследования эффективности. Важно заметить, что при одинаковых требованиях к измельчению (одинаковом критерии измельчения) количество ячеек расчетной сетки для предложенного решения больше, чем при использовании классического ячейко-ориентированного подхода, однако необходимый объем памяти для хранения сетки в памяти процессора меньше, так как нет необходимости в сохранении указателей на соседей каждой ячейки.

Таблица 1 Сравнение построенных расчетных сеток

	Подход p4est	*Предложенный подход*
Базовый уровень ячеек	9	9
Максимальный уровень ячеек	12	12
Базовый уровень блоков	–	3
Максимальный уровень блоков	–	6
Общее количество ячеек	851968	721768
Общее количество блоков	–	208
Объем памяти (байты)	52232064	20447232
Время построения (с)	1,66	0,96

5.3 Результаты экспериментов

В ***Таблица 2*** представлены результаты запусков на различном числе процессоров с заданным количеством потоков на суперкомпьютере BlueGene/P, значение количества потоков на каждый процесс равно 4. В ***Таблица 3*** представлены результаты запусков на различном числе процессоров с заданным количеством потоков на суперкомпьютере BlueGene/P, значение количества потоков на каждый процесс равно 4. На ***Рисунок 8*** показаны эффективность гибридного распараллеливания предложенного решения для

различного количества блоков и эффективность гибридного распараллеливания ячейко-ориентированного подхода. На ***Рисунок 9*** приведено сравнение времени выполнения расчетов с применением различных стратегий представления сеток и распределения вычислительной нагрузки.

Таблица 2 Результаты экспериментов по исследованию эффективности предложенного подхода (BlueGene/P, режим SMP, 4 потока на 1 процесс, 1000 шагов по времени)

Количество блоков	*Количество процессов* •	*Время вычисления температур (максимум по процессам) (с)*	*Время обменов и ожидания обменов (максимум по процессам) (с)*	*Общее время вычислений (максимум по процессам) (с)*	*Ускорение*	*Эффективность*
208	1	848,579	0,000	848,579	1,000	1,000
	2	454,458	69,923	454,578	1,867	0,933
	4	260,565	71,577	260,663	3,255	0,814
	8	131,568	37,501	131,633	6,447	0,806
	16	68,447	22,045	68,564	12,376	0,774
832	1	1 029,100	0,000	1 029,100	1,000	1,000
	2	537,295	63,741	537,513	1,915	0,957
	4	299,377	70,814	299,523	3,436	0,859
	8	152,955	40,435	153,158	6,719	0,840
	16	79,201	24,103	79,326	12,973	0,811
3148	1	1 372,650	0,000	1 372,650	1,000	1,000
	2	702,532	63,628	702,960	1,953	0,976
	4	376,979	69,360	377,242	3,639	0,910
	8	189,263	40,286	189,655	7,238	0,905
	16	97,573	26,519	97,796	14,036	0,877

Таблица 3 Результаты экспериментов по исследованию эффективности подхода p4est (BlueGene/P, режим SMP, 4 потока на 1 процесс, 1000 шагов по времени)

Количество процессов	*Время вычисления температур (максимум по процессам) (с)*	*Время обменов и ожидания обменов (максимум по процессам) (с)*	*Общее время вычислений (максимум по процессам) (с)*	*Ускорение*	*Эффективность*
1	577,426	0,000	2 887,145	1,000	1,000
2	1 467,800	205,707	1 475,810	1,956	0,978
4	735,745	107,442	149,272	3,868	0,967
8	367,699	57,491	746,360	7,715	0,964
16	183,688	30,904	189,483	15,237	0,952

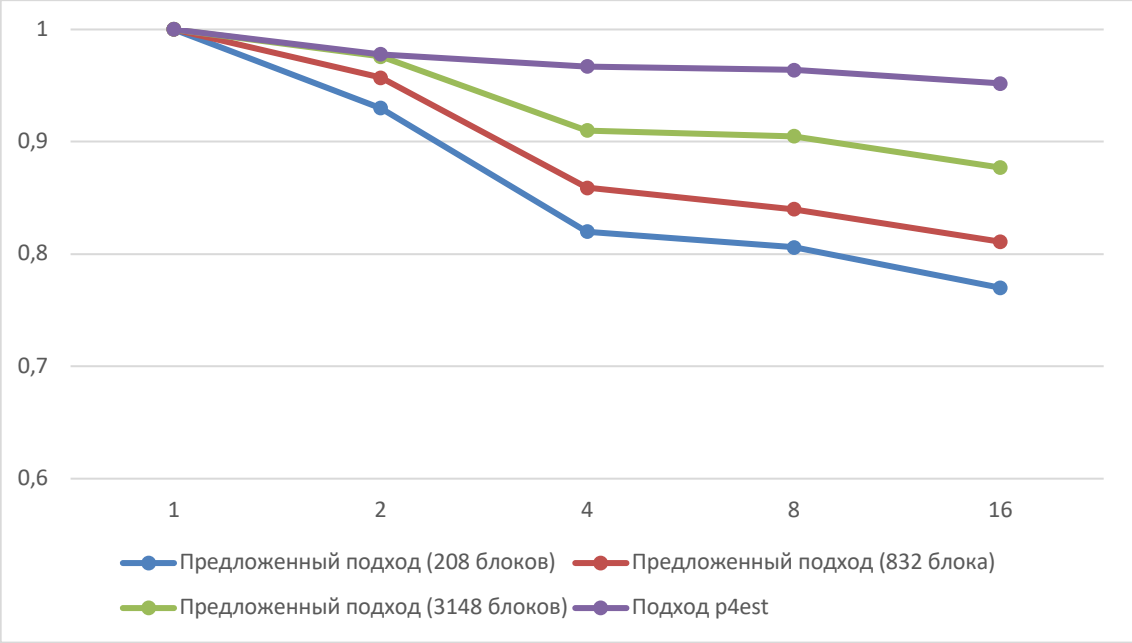

Рисунок 8 Графики эффективности распараллеливания предложенного подхода для различных значений блоков и подхода p4est

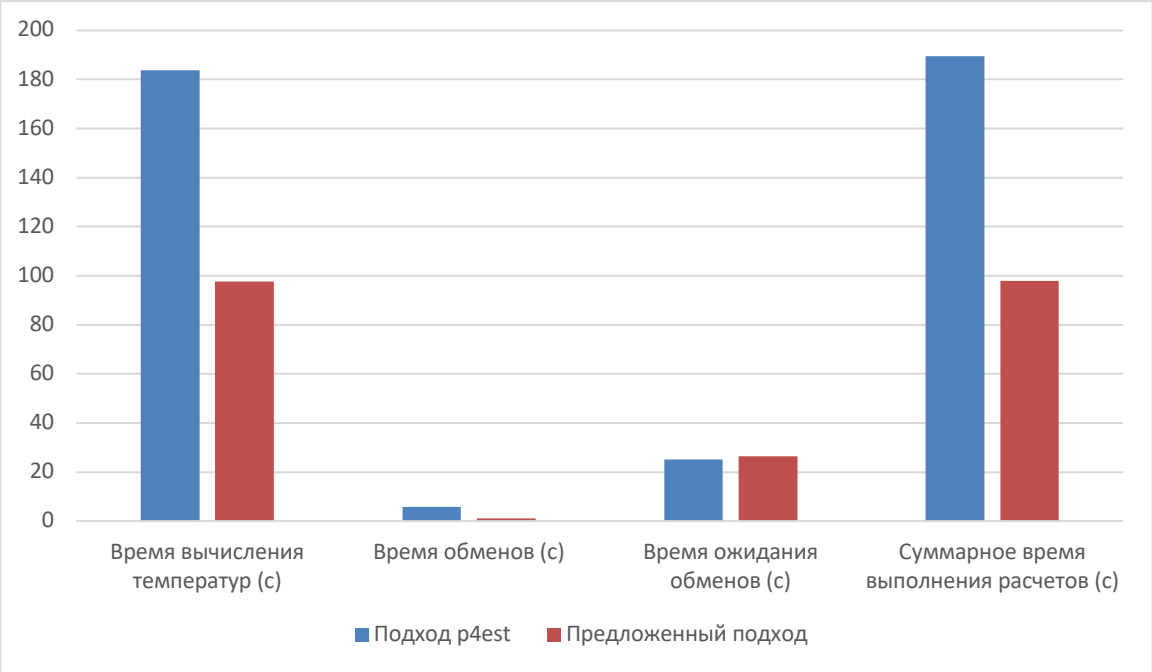

Рисунок 9 Графики сравнения времени выполнения расчетов на основе подхода p4est и предложенного подхода

5.4 Выводы

Экспериментальное исследование показало высокую эффективность предложенного решения, а также его преимущества перед подходом p4est в использовании оперативной памяти при использовании для метода конечных объемов.

В ходе экспериментов было замечено, что несмотря на хороший баланс по весам блоков, при расчетах возникает дисбаланс расчетов значений температур в ячейках. Причем время обработки границ блоков на всех процессах примерно одинаковое, а различается время расчетов именно во внутренних ячейках. Вероятно, это связано с тем, что размеры блоков различаются, то есть имеют различную эффективность использования кэша. Влияние данной особенности снижается с увеличением числа процессоров.

Исследование показали, что эффективность распараллеливания в большей степени зависит от равномерности распределения нагрузки, так как время, затрачиваемое на обмены мало по сравнению с временем ожидания обменов. Поэтому с увеличением количества блоков эффективность распараллеливания растет (***Рисунок 8***). Однако с увеличением количества блоков эффективность использования вычислительных узлов падает, что видно из увеличения времени выполнения расчетов температур (***Таблица 2***). По сравнению с подходом p4est, в предложенном подходе время непосредственно обменов меньше за счет более ровных границ доменов, то есть более быстрой упаковки данных, однако время ожидания обменов немного больше из-за большего дисбаланса вычислительной нагрузки.

Время, затрачиваемое на выполнение расчетов значений температур для предложенного решения существенно меньше, чем для ячейко-ориентированного подхода. Это объясняется тем, что при обходе блоков используется регулярный и локальный шаблон доступа к памяти, что увеличивает эффективность использования кэша. Более того, объем памяти, занимаемый сеткой, существенно меньше (***Таблица 1***), что опять же увеличивает эффективность использования кэша.

Таким образом, основными результатами является:

1. Обработка домена расчетной сетки внутри вычислительного узла быстрее почти в 2 раза за счет эффективного использования кэша.

2. Время обменов данными между вычислительными узлами быстрее более чем в 5 раз быстрее за счет более регулярной формы границ доменов.

3. Эффективность распараллеливания сравнима с ячейко-ориентированным подходом, может быть увеличена с помощью увеличения количества блоков.

6. Заключение

В ходе проведенной работы были достигнуты следующие результаты.

1. Предложена структура данных для представления декартовых локально-сгущающихся сеток: дерево блоков в компактном представлении.

2. Разработаны гибридные алгоритмы декомпозиции расчетной сетки в данном представлении на домены и обработки доменов внутри вычислительного узла.

3. Проведено экспериментальное исследование эффективности предложенного решения, показана эффективность предложенного подхода при использовании преимущество предложенного решения перед подходом p4est для модельной двумерной задачи.

Исследование показало, что предложенный подход может дать преимущество перед распространенными на данный момент в России подходами и может быть использован в текущих исследованиях. Развитие данной работы может происходить в сторону обобщения решения на трехмерный случай, оптимизации размеров блоков для конкретных задач, реализации различных численных схем с использованием предложенного подхода.

7. Список литературы

1. Абалакин И. В., Жданова Н. С., Суков С. А. Реконструкция геометрии объекта на элементах неструктурированной сетки при использовании метода погруженных границ //Математическое моделирование. – 2016. – Т. 28. – №. 6. – С. 77-88.

2. Афендиков А. Л. и др. Алгоритм многоуровневой адаптации сеток по критериям на основе вейвлетанализа для задач газовой динамики //Препринты Института прикладной математики им. МВ Келдыша РАН. – 2015. – №. 0. – С. 97-22.

3. Афендиков, А. А. Давыдов, А. Е. Луцкий и др. АДАПТИВНЫЕ ВЕЙВЛЕТНЫЕ АЛГОРИТМЫ ДЛЯ РЕШЕНИЯ ЗАДАЧ ГИДРО И ГАЗОВОЙ ДИНАМИКИ НА ДЕКАРТОВЫХ СЕТКАХ. //М.: ИПМ им. М.В.Келдыша Москва, 2016.

4. Горобец А. В., Суков С. А., Триас Ф. Х. Проблемы использования современных суперкомпьютеров при численном моделировании в гидродинамике и аэроакустике //Ученые записки ЦАГИ. – 2010. – Т. 41. – №. 2.

5. Дерюгин Ю. Н., Саразов А. В., Жучков Р. Н. Особенности построения методики расчёта на сетках типа «Химера» для неструктурированных сеток //Математическое моделирование. – 2017. – Т. 29. – №. 2. – С. 106-118.

6. Лисейкин В. Д. Разностные сетки. Теория и приложения //Новосибирск, издательство СО РАН. – 2014.

7. Лисейкин В. Д. и др. Технология построения разностных сеток. – 2009.

8. Меньшов И. С., Корнев М. А. Метод свободной границы для численного решения уравнений газовой динамики в областях с изменяющейся геометрией //Математическое моделирование. – 2014. – Т. 26. – №. 5. – С. 99-112.

9. Самарский А. А. Теория разностных схем. – " Наука," Глав. ред. физико-математической лит-ры, 1989.

10. Самарский А. А., Гулин А. В. Численные методы математической физики. – М : Науч. мир, 2003.

11. Смирнов Е. М., Зайцев Д. К. Метод конечных объемов в приложении к задачам гидрогазодинамики и теплообмена в областях сложной геометрии //Научно-технические ведомости. – 2004. – №. 2. – С. 70-81.

12. Суков С. А. Методы генерации тетраэдральных сеток и их программные реализации //Препринты Института прикладной математики им. МВ Келдыша РАН. – 2015. – №. 0. – С. 23-22.

13. Сухинов А. А. Построение декартовых сеток с динамической адаптацией к решению //Математическое моделирование. – 2010. – Т. 22. – №. 1. – С. 86-98.

14. Тихонов А. Н., Самарский А. А. Уравнения математической физики. 3-е изд., стер. – 1989.

15. Якобовский М. В. Введение в параллельные методы решения задач //М.: МГУ. – 2013.

16. Adams M. et al. Chombo software package for amr applications-design document. – 2015. – №. LBNL6616E.

17. Aftosmis M. J. Solution adaptive Cartesian grid methods for aerodynamic flows with complex geometries //VKI Lecture Series. – 1997. – Т. 2. – С. 1997.

18. Berger M. J., Colella P. Local adaptive mesh refinement for shock hydrodynamics //Journal of computational Physics. – 1989. – Т. 82. – №. 1. – С. 64-84.

19. Berger M., Rigoutsos I. An algorithm for point clustering and grid generation //IEEE Transactions on Systems, Man, and Cybernetics. – 1991. – Т. 21. – №. 5. – С. 1278-1286.

20. 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. – Т. 33. – №. 3. – С. 1103-1133.

21. Dubey A. et al. A survey of high level frameworks in block-structured adaptive mesh refinement packages //Journal of Parallel and Distributed Computing. – 2014. – Т. 74. – №. 12. – С. 3217-3227.

22. Hannoun N., Alexiades V. Issues in adaptive mesh refinement implementation //Electronic Journal of Differential Equations. – 2007. – Т. 15. – С. 141-151.

23. Hariharan N. et al. Enabling radiative transfer on AMR grids in crash //Monthly Notices of the Royal Astronomical Society. – 2017. – Т. 467. – №. 2. – С. 2458-2475.

24. Karypis G., Kumar V. Multilevelk-way partitioning scheme for irregular graphs //Journal of Parallel and Distributed computing. – 1998. – Т. 48. – №. 1. – С. 96-129.

25. Salem F. K. A., Arab M. A. Comparative study of space filling curves for cache oblivious TU Decomposition //arXiv preprint arXiv:1612.06069. – 2016.

26. Schloegel K., Karypis G., Kumar V. Parallel static and dynamic multi-constraint graph partitioning //Concurrency and Computation: Practice and Experience. – 2002. – Т. 14. – №. 3. – С. 219-240.

27. Teyssier R. Cosmological hydrodynamics with adaptive mesh refinement-A new high resolution code called RAMSES //Astronomy & Astrophysics. – 2002. – Т. 385. – №. 1. – С. 337-364.