МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ
Харьковский национальный университет имени В. Н. Каразина

На правах рукописи

Паточкин Борис Викторович

УДК 519.6

МОДЕЛИРОВАНИЕ ДИФРАКЦИОННЫХ ПРОЦЕССОВ НА ОСНОВЕ МЕТОДОВ ДИСКРЕТНЫХ ОСОБЕННОСТЕЙ ПРИ ОГРАНИЧЕННЫХ ВЫЧИСЛИТЕЛЬНЫХ РЕСУРСАХ

01.05.02 – Математическое моделирование и вычислительные методы

Диссертация на соискание ученой степени
кандидата технических наук

Научный руководитель
Мищенко Виктор Олегович
доктор технических наук, профессор

Харьков – 2016

СОДЕРЖАНИЕ

СПИСОК ИСПОЛЬЗОВАННЫХ СОКРАЩЕНИЙ

ГСИУ	гиперсингулярное интегральное уравнение
МДО	метод(ы) дискретных особенностей
МДТ	метод(ы) дискретных токов
ПК	персональный компьютер
ПС	программная система
СЛАУ	система линейных алгебраических уравнений
СИУ	сингулярное интегральное уравнение
СМЧР	стандартное минимальное численное решение
AVX	advanced vector extensions
FMA	fused multiply-add
TDP	thermal design point

ВВЕДЕНИЕ

Актуальность темы. При всестороннем исследовании антенн, рефлекторов, открытых волноводов и т.д. существует потребность в моделировании дифракции электромагнитных волн на системах металлических экранов для многих вариантов формы экранов и различных волновых чисел. При изучении механических конструкций также часто возникает потребность в исследовании дифракционных явлений в материалах и элементах этих конструкций при взаимодействии упругих волн с технологическими отверстиями, включениями, а также дефектами. Все это требует многовариантных расчетов, и время компьютерного моделирования становится критическим параметром успешности своевременного завершения конструкторских работ или исследования состояния существующего оборудования. Традиционный способ преодоления этой проблемы - использование параллельных вычислений с увеличением степени физического распараллеливания по мере увеличения сложности прикладных задач. Соответственно растут и требования к скорости доступа к основной памяти компьютеров. Это главные вычислительные ресурсы компьютера. Однако не всегда есть финансовые и организационные возможности для удовлетворения этих требований по каждому поводу, и большая часть инженеров и исследователей, если не их большинство, постоянно работают только на персональных компьютерах (ПК), где указанные параметры ощутимо ограничены.

Резервы ускоренного моделирования сложных явлений на ПК остается искать в архитектуре этих систем, в современных условиях достаточно совершенной и сложной. Общих решений ожидать нельзя, но чем конкретней класс задач рассматривается, тем больше шансов, что найдутся специальные вычислительные методы решения этих задач, позволяющие использовать дополнительные вычислительные мощности, скрытые в архитектуре.

Поэтому, возвращаясь к проблемам моделирования дифракционных явлений сразу ограничимся строгими методами (то есть такими, которые не используют априорных предположений о возможном упрощении, при определенных условиях, того явления, которое моделируется). Среди современных строгих математических методов приближенного решения дифракционных задач сейчас пользуются большой популярностью методы дискретных особенностей (МДО), которые являются методами построения дискретных моделей граничных псевдодиференциальних уравнений дифракционной задачи, если те получены. Обычно эти методы обеспечивают концептуальную простоту метода приближенного решения задачи и его достаточную обусловленность. Еще более узкий класс составляют методы дискретных токов (МДТ), отличающиеся прозрачностью и естественностью при моделировании взаимодействия электромагнитных волн с проводящими экранами.

Подытоживая сказанное, можно сделать вывод, что тема диссертационной работы имеет важное научное и практическое значение, поскольку она направлена на решение *актуальной научно-прикладной задачи*, сущность которой заключается в обеспечении основанного на МДО моделировании дифракционных процессов на ПК вычислительными методами, которые наряду с простым распараллеливанием вычислений на процессорные ядра, позволяют использовать дополнительные вычислительные мощности, скрытые в архитектуре современных ПК, без особых усилий программирования и дополнительных затрат энергии при выполнении.

Связь работы с научными программами, планами, темами. Диссертационная работа выполнена на кафедре моделирования систем и технологий факультета компьютерных наук в Харьковском национальном университете им. В. Н. Каразина в соответствии с планом подготовки аспиранта и планов бюджетных научно-исследовательских работ по темам МОН Украины: – «Математическое моделирование физических процессов и

численный эксперимент» (ДР 01040002366) и «Методы дозиметрии электронного излучения с применением компьютерного моделирования» (ДР 0109U001436).

Цель и задачи исследования. Цель исследования - обеспечить выигрыш во времени моделирования дифракционных процессов методами дискретных особенностей, превосходящий максимально возможную степень распараллеливания вычислений на процессорные ядра компьютера.

Для достижения этой цели были определены следующие *задачи исследования*:

1) Уточнить направления исследований на основе анализа источников по теме.

2) Разработать метод ускорения компьютерного решения дифракционных задач, основанного на МДО, на ПК за счет согласования схемы расчетов с иерархической организацией памяти этих ПК.

3) Найти способ ускорения моделирования дифракционных процессов, использующих МДО, на ПК за счет дополнительных вычислительных мощностей современных процессоров.

4) Обеспечить в компьютерном моделировании дифракционных процессов с помощью МДО согласованность таких конкурирующих требований, как рациональность использования компьютерных ресурсов, скорость вычислений и возможность реализации первых двух требований умеренными усилиями прикладного программиста (знакомого с разработанными в диссертации вычислительными методами).

5) Разработать метод контроля качества моделирующих программных систем, реализующих ускоренные вычислительные методы диссертации, по характеристике энергопотребления.

6) Протестировать и передать заинтересованным организациям программные системы, которые реализуют разработанные вычислительные методы моделирования дифракционных процессов на ПК.

Объектом исследования является процесс компьютерного

моделирования дифракционных явлений на основе параллельных вычислительных методов дискретных особенностей.

Предметом исследования является моделирование дифракционных явлений методами дискретных особенностей, преобразованными для выполнения на персональных компьютерах с расчетом на обеспечение скорости исполнения, превосходящей предел, в общем случае накладываемый ограничениями главных вычислительных ресурсов.

Методы исследования. Теоретической основой исследования является теория МДО и компьютерной математики. В процессе анализа использования компьютерной памяти вычислительными методами класса МДО использована техника комбинаторного перебора с отсечением ветвей. При выяснении значений квазиоптимальных параметров для вычислительных методов применялся спланированный численный эксперимент на множестве доступных ПК. При поиске квазиоптимальных действий с векторными регистрами использованы оптимизирующие свойства компиляторов и дизассемблирование. Эмпирические зависимости устанавливались методами регрессии. Для проверки нужных свойств компьютерных реализаций вычислительных методов привлекались метрические методы оценки качества программных систем (ПС).

Научная новизна полученных результатов:

1) Впервые разработан тайлинговый параллельный вычислительный метод 2D моделирования дифракции электромагнитных волн на проводящих экранах, который относится к группе МДТ, однако отличается от известных методов, считая параллельные, схемой вычислений, которая позволяет улучшить локальность памяти при реализации в мультиядерной системе одного или нескольких ПК, существенно уменьшая время расчета матрицы, решения системы линейных алгебраических уравнений (СЛАУ) дискретной модели граничных уравнений, вычисления диаграмм направленности, и предназначена служить шаблоном аналогичных решений для других МДО.

2) Усовершенствован МДТ в направлении его векторизации,

которая позволяет применить AVX команды процессора, если они присутствуют в ПК, с целью использования векторных регистров для уменьшения времени расчета матрицы, решения СЛАУ дискретной модели граничных уравнений и вычисления диаграмм направленности, причем указан прозрачный метод переноса этой векторизации на другие МДО решения дифракционных задач.

3) Усовершенствован метод организации параллельного компьютерного моделирования дифракционных процессов, основанный на граничных интегральных уравнениях и МДО, в направлении достижения средствами высокоуровневого программирования согласованности критериев динамичности распределения ресурсов компьютера, сохранения скорости вычислений и простоты программной реализации, что позволяет, в частности, в достаточной степени реализовать потенциал быстрого выполнения разработанных в диссертации вычислительных методов, не требуя от прикладника-реализатора решения сложных задач системного программирования.

4) Впервые разработан метод проверки качества ПС компьютерного моделирования по характеристике расхода электрической энергии, аналогов которого по публикациям не найдено, который имеет собственные метрики, примитивы которых получаются программными средствами и из документации, позволяя без специального оборудования не только проверять отсутствие перерасхода энергии, в частности при использовании созданных в диссертации моделирующих ПС, но и сравнивать расходы различных методов и даже оценивать достоверное уменьшение расходов лучших методов при их применении.

Практическая ценность полученных результатов. Основные научные результаты диссертации внедрены при решении задач ускорения численного моделирования процессов взаимодействия волн различной природы с рассеивателями при исследовании технических устройств и материалов.

В ИПМ им. А.М. Подгорного НАН Украины при выполнении работ по бюджетной теме №29 для своевременного расчета достаточного количества вариантов, которые предназначены для рассмотрения при получении уточненных прогнозов относительного продления сроков работы нефте-химического оборудования (справка о внедрении от 01.11.2016).

В Институте телекоммуникаций и глобального информационного пространства НАН Украины тайлинговий вычислительный метод 2D моделирования дифракции упругих волн, метод переноса приемов векторизации между различными вычислительными методами и новый метод организации параллельного компьютерного моделирования дифракционных процессов были внедрены в рамках комплекса научно-исследовательских работ по теме «Разработка вычислительных технологий и методов моделирования для исследования нестационарных процессов» (акт внедрения от 15.10.2016).

Научные результаты диссертации нашли также внедрение в Харьковском национальном университете имени В.Н. Каразина в учебном процессе в дисциплинах курса подготовки бакалавров «Технология распределенных систем и параллельных вычислений», «Численные методы», «Системы контроля и управления» и в дисциплине подготовки магистров «Масштабируемые программные системы» (акт внедрения от 15.11.2016).

Личный вклад соискателя. Диссертационная работа задумана и выполнена лично автором. Все научные результаты соискателем получены самостоятельно. Четыре работы [65-68] опубликованы без соавторов. В опубликованных в соавторстве научных трудах соискателю принадлежит: в [45] - аналитический отсев вариантов построения модифицированного метода решения системы линейных уравнений, уточнение значений параметров метода и экспериментальная проверка выводов; в [49, 51, 53, 64, 96] - все новые научные результаты, не считая постановок задач и общих концептуальных положений, которые принадлежат соавтору (научному руководителю); в [52] - разработка и получение результатов тестирования

одного из вариантов реализации программы.

Апробация результатов диссертационной работы. Основные результаты диссертации докладывались и обсуждались на конференциях:

- Международная научно-техническая конференция «Компьютерное моделирование в наукоемких технологиях» 24-27 апреля 2012 г., Харьковский национальный университет им. В.Н. Каразина;

- Международная школа-конференция «Современные проблемы математики, механики и информатики» 29 сентября - 4 октября 2013, Харьковский национальный университет им. В.Н. Каразина;

- Международная научно-техническая конференция «Компьютерное моделирование в наукоемких технологиях» 28-31 апреля 2014, Харьковский национальный университет им. В.Н. Каразина;

- 17-й международный симпозиум «Методы дискретных особенностей в задачах математической физики». 8-13 июня 2015, Сумы, Сумской государственный университет;

- Международная научная конференция «Современные проблемы моделирования и вычислительных методов». 19-22 февраля 2015, Ровно, Ровенский государственный гуманитарный университет.

- 12-я Международная научно-практическая конференция «Теоретические и прикладные аспекты построения программных систем» (TAAPSD-2015), Киев, 23-25 ноября 2015 года на базе национального университета «Киево-Могилянская Академия» (доклад включена в опубликованной программы конференции) .

В полном объеме диссертация докладывалось на научном семинаре «Методы дискретных особенностей в задачах математической физики» ХНУ им. В.Н. Каразина и на расширенном научном семинаре кафедры моделирования систем и технологий ХНУ им. В.Н. Каразина.

Публикации. По теме диссертационной работы опубликовано 11 научных работ, среди которых 4 без соавторов, 5 статей в профессиональных изданиях по специальности, из них - 1 статья в издании, которое включено в

справочник периодических изданий базы данных Ulrich's Periodicals Directory (New Jersey, USA) [51], есть сертификат авторского права на компьютерную программу [1].

РАЗДЕЛ 1
АНАЛИЗ ИСТОЧНИКОВ ПО ТЕМЕ ДИССЕРТАЦИИ

Компьютерное моделирование дифракционных процессов на основе специальных дискретных моделей граничных интегральных и псевдодифференциальных операторов является востребованной и хорошо соложившиейся областью прикладных исследований. Используемые при этом вычислительные методы, которые относятся к классу МДО, полностью обоснованы в 2D постановке в смысле сходимости приближенных решений, получаемых из дискретных моделей, к точному решению краевой задачи при стремлении параметра дискретизации (определяющего размерность дискретной модели) к ∞. Однако, это – традиционные методы в том смысле, что подразумевают в качестве исполнителя абстрактную машину последовательного действия с неограниченной памятью. В последние 10-20 лет появлялись также примеры параллельных МДО, получаемых преобразованиями традиционных МДО с целью их ориентации на выполнение коллективом исполнителей, поддерживающих свои последовательные вычислительные процессы, прерываемые сигналами синхронизации и обменом данными. Цель этой работы имеет истоком то обстоятельство, что параллельные МДО не позволяют ускорить численное моделирование дифракционных процессов (в сравнении с традиционными методами) в число раз, большее количества имеющихся физических устройств, которые выполняют отдельные вычислительные процессы. В ПК такие устройства – ядра процессоров, и их число существенно ограничено обычно 2-6). Преодолеть это ограничение для МДО можно только, используя одновременно какую-то их специфику и подходящие особенности особенности архитектуры ПК. Поэтому в данном разделе большое внимание уделяется вычислительной схеме и расчётным формулам МДО, а также резервам повышения скорости выполнения программ на ПК. Материал основан на обзорах автора в публикациях [45, 49, 65-67].

1.1Задачи дифракции электромагнитных волн на проводящих экранах в двумерной постановке.

Постановки краевых задач математической теории дифракции, основывается на физических законах, управляющих распространением звука, электромагнитных или упругих волн и их взаимодействия с препятствиями. Если в задаче присутствует симметрия, то её постановка редуцируется, например, 2D постановки возникают вследствие сдвиговой симметрии.

Рассмотрим пример электродинамики, к слову, хорошо отражающий черты дифракционных процессов другой природы. Первоначально вычислительные методы решения таких задач, как правило, основывались на их сведении к системе интегральных уравнений второго рода с ограниченными интегральными операторами, действующими в подходящих пространствах функций на границах областей, заполненных диэлектриком [34]. В 80-е годы прошлого века точка зрения на целесообразность базирования вычислительных методов для дифракционных задач на таких операторах изменилась. Это произошло благодаря осмыслению эвристического метода дискретных вихрей Белоцерковского как вычислительного метода решения сингулярного интегрального уравнения первого рода [6]. Тогда возникн общий подход к решению краевых задач математической физики, в том числе, дифракционных, с использованием методов дискретных особенностей [14, 37].

Математически строгим воплощением этого подхода является метод дискретных токов Ю. В. Ганделя [14, 15, 17] и его метод параметрических представлений интегральных и псевдодифференциальных операторов [18, 21, 22].

Известны разные варианты МДТ, в том числе основанные на физических соображениях [31]. Изложим схему такого метода, преобразования которого привели к вычислительным методам диссертации. Изложение следует содержанию работ [20, 19, 31], но обозначения частично изменены ради единообразия.

Пусть в плоской задаче дифракции электромагнитных волн на идеально проводящих контурах зависимость от времени задается множителем $e^{i\omega t}$. Амплитуды наблюдаемого полного поля $\vec{E}^{tot} = \vec{E} + \vec{E}_0$, $\vec{H}^{tot} = \vec{H} + \vec{H}_0$, где $\vec{E}_0, \vec{H}_0$ - падающее поле, E, H - рассеянное, причем

Для случая E-поляризации, падающее поле представлено в виде

$$E_0 = (0,0,U_0),$$

$$H_0 = \left(\frac{1}{i\omega\mu} \frac{\partial U_0}{\partial y}, \frac{1}{i\omega\mu} \frac{\partial U_0}{\partial x}, 0 \right).$$

Можно, как хорошо известно, считать, что полное поле

$$E^{tot} = (0,0,U^{tot}),$$

$$U^{tot}(X) = U(X) + U_0(X),\ X \in R^2.$$

Тогда компонента рассеянног поля U удовлетворяет уравнению Гельмгольца всюду вне множества $C = \bigcup_{n=0}^{N-1} C_n$, где C_n – n-контур, а N - количество контуров:

$$\begin{aligned} &\Delta U(X) + k^2 U(X) = 0, \\ &x \in R^2 / \bar{C}, X = (x,y). \end{aligned} \tag{1.1}$$

На контуре C выполняется условие Дирихле

$$U(X)\big|_c = -U_0(X)\big|_c, \tag{1.2}$$

на бесконечности - условия излучения Зоммерфельда

$$U(X) \underset{|X|\to\infty}{=} O\left(\frac{1}{\sqrt{r}} \right), r = |X|, \tag{1.3}$$

$$\frac{\partial U(X)}{\partial r} - ikU(X) \underset{x\to\infty}{=} O\left(\frac{1}{r^{3/2}} \right), \tag{1.4}$$

а на всех концах контуров - условие на ребре

$$\int_\Omega \left(|U(X)|^2 + |\nabla U(X)|^2 \right) d\Omega < \infty, \tag{1.5}$$

где Ω это ограниченное открытое множество, содержащее конец контура.

В диссертации используются интегральные уравнения первого рода с логарифмической особенностью, которое предусматривает меньше машинных вычислений, чем СИУ.

Для получения рабочих формул решение задачи (1.1)-(1.5) часто ищут [20, 31, 27] в виде потенциала простого слоя:

$$U(Y)=\int_C \mu(X)H_0^{(2)}\left(k\left|X-Y\right|\right)dl_X\,, \tag{1.6}$$

где $\mu(X)$ - линейная плотность простого слоя в точке $X=(x_1,x_2)\in C$.

Формальная подстановка (1.6) в граничное условие (1.2) приводит к интегральному уравнению на системе контуров (ИУ):

$$\int_C \mu(X)H_0^{(2)}\left(k\left|X-Y\right|\right)dl_X=-U_0(Y),X,Y\in C \tag{1.7}$$

Если ввести индексацию и параметризацию контуров $C_n, n=0,...,N-1,$

$$C_n:\begin{cases}x_1=x_1^{(n)}(t)\\ x_2=x_2^{(n)}(t)\end{cases}, \tag{1.8}$$

ИУ (1.7) становится эквивалентно системе интегральных уравнений на отрезках:

$$\sum_{n=0}^{N-1}\int_{-1}^{1}\mu^{(n)}(t)\mathrm{K}^{(n,m)}(t,\tau)L^{(n)}(t)dt=-U_0\left(x_1^{(m)}(\tau),x_2^{(m)}(\tau)\right),\ m=0,...,N-1, \tag{1.9}$$

где

$$\mu^{(n)}=\mu_{\big|_{C_n}},\ \mu^{(n)}(t)=\mu\left(x_1^{(n)}(t),x_2^{(n)}(t)\right),$$

$$\mathrm{K}^{(n,m)}(t,\tau)=H_0^{(2)}\left(k\sqrt{\sum_{p=1}^{2}\left(x_p^{(n)}(t)-x_p^{(m)}(\tau)\right)^2}\right),$$

$$L^{(n)}(t)=\sqrt{\left(x_1^{(n)}(t)'\right)^2-\left(x_2^{(n)}(t)'\right)^2}\,,$$

$$H_0^{(2)}(z)\underset{z\to 0}{=}1-\frac{2i}{\pi}\left[\ln\frac{z}{2}+\gamma\right]+O\left(z^2\ln z\right). \tag{1.10}$$

Ядра уравнений (9) имеют поведение в нуле

$$H_0^{(2)}\left(k\sqrt{\sum_{p=1}^{2}\left(x_p(t)-x_p(\tau)\right)^2}\right)\underset{\tau\to t}{=}a(t)+b(t)\ln|\tau-t|+O\left((\tau-t)^2\right) \qquad (1.11)$$

определяющее логарифмический характер особенности ядра:

$$a(t)=1-\frac{2i}{\pi}\left[\frac{1}{2}\ln\left(\sum_{p=1}^{2}\left(x_p'(t)\right)^2\right)+\ln\frac{k}{2}+\gamma\right], \qquad (1.14)$$

$$b(t)=-\frac{2i}{\pi}, \qquad (1.15)$$

Полагая

$$u^{(n)}(t)=\sqrt{1-t^2}\,\mu^{(n)}(t)\cdot L^{(n)}(t), \qquad (1.16)$$

где $\mu^{(n)}(t)\in \mathrm{L}_{2,\rho}$, $\rho(t)=\sqrt{1-t^2}$.

Удобно придать системе (1.9) вид:

$$\frac{-2i}{\pi}\int_{-1}^{1}\ln|\tau-t|\frac{u^{(m)}(t)}{\sqrt{1-t^2}}dt+\sum_{n=0}^{N-1}\int_{-1}^{1}\frac{u^{(n)}(t)}{\sqrt{1-t^2}}K^{(m,n)}(\tau,t)dt=f^{(m)}(\tau), \qquad (1.17)$$

где $f^{(m)}(\tau)=-U_0\left(x_1^{(m)}(\tau),x_2^{(m)}(\tau)\right)$,

$$\begin{aligned}K^{(m,n)}(\tau,t)&=K_1^{(m,n)}(\tau,t),\ m\neq n,\\ K^{(n,n)}(\tau,t)&=K_1^{(n,n)}(\tau,t)+\frac{2i}{\pi}\ln|\tau-t|,\end{aligned} \qquad (1.18)$$

где $K_1^{(m,n)}(\tau,t)=H_0^{(2)}\left(k\sqrt{\sum_{p=1}^{2}\left(x_p^{(n)}(t)-x_p^{(m)}(\tau)\right)^2}\right)$.

Для слуая Н-поляризации, падающее поле представлено в виде

$$H_0=(0,0,U_0),$$

$$E_0=\left(-\frac{1}{i\omega\varepsilon}\frac{\partial U_0}{\partial y},\frac{1}{i\omega\varepsilon}\frac{\partial U_0}{\partial x},0\right). \qquad (1.19)$$

Используем выражения электрического и магнитного поля через скалярные ψ и векторные $\vec{A}$ потенциалы

$$\vec{E} = -grad\psi - i\omega\vec{A},$$
$$\vec{H} = \frac{1}{\mu} rot\vec{A}. \qquad (1.20)$$

Выражение векторного и скалярного (1.20) потенциала через токи на поверхности

$$\vec{A}(N) = \frac{\mu}{4\pi}\int_S \vec{j}(M)\frac{\exp\{-i\kappa L_{MN}\}}{L_{MN}} ds_M, \qquad (1.21)$$

$$\psi(N) = \frac{i}{4\pi\omega\varepsilon}\int_S \vec{j}(M)\cdot grad\left\{\frac{\exp\{-i\kappa L_{MN}\}}{L_{MN}}\right\} ds_M. \qquad (1.22)$$

Тангенциальная составляющая полного электрического поля обращается на поверхности в ноль.

$$\left[\vec{E}^0, \vec{n}^0\right] = -\left[\vec{E}, \vec{n}^0\right],\ M \in S. \qquad (1.23)$$

Пусть $\vec{\tau}^0$ касательная к контуру тогда граничное условие (1.23) запишется в следующем виде

$$\frac{1}{h_\tau}\frac{\partial\psi}{\partial\tau} + i\omega A_\tau(N) = E_\tau^0,\ \ q = q_0,\ \tau \in [0, 2\pi). \qquad (1.24)$$

Наведенный ток в случае Н-поляризации имеет только τ составляющую

$$\vec{j} = \vec{t}^0 j_t(t),\ t \in [0, 2\pi) \qquad (1.25)$$

с учетом (1.25) скалярный потенциал (1.22) представляется в виде (1.26), а векторный потенциал (1.121) в виде (1.29)

$$\psi(N) = \frac{1}{4\omega\varepsilon}\int_0^{2\pi} j_\tau(t)\cdot\frac{\partial}{\partial t} H_0^{(2)}(\kappa L_{MN}) dt, \qquad (1.26)$$

$$A_\tau(N) = \frac{\mu}{4h_\tau i}\int_0^{2\pi} h(\tau,t)\, H_0^{(2)}(\kappa L_0)\, j_\tau(t) dt, \qquad (1.27)$$

$$h(\tau,t) = \frac{\partial x_1(t)}{\partial t}\frac{\partial x_1(\tau)}{\partial\tau} + \frac{\partial x_2(t)}{\partial t}\frac{\partial x_2(\tau)}{\partial\tau}. \qquad (1.28)$$

Подставляя (1.26, 1.27) в граничное условие (1.24) получается следующее интегро-дифференциальное уравнение

$$\lim_{q\to q_0}\frac{\partial}{\partial\tau}\int_0^{2\pi}\frac{\partial}{\partial t}H_0^{(2)}(\kappa L)\,j_\tau(t)\,dt-k^2\int_0^{2\pi}h(\tau,t)\,H_0^{(2)}(\kappa L_0)\,j_\tau(t)dt=$$
$$=4h_\tau(\tau)E_\tau^0(\tau) \qquad (1.29)$$

$$L=L(q,\tau,t)=\left\{\left(x(q,\tau)-\xi(t)\right)^2+\left(y(q,\tau)-\eta(t)\right)^2\right\}^{\frac{1}{2}},\ L_0=L(q_0,\tau,t),$$

$$h(\tau)=\left[\frac{\partial x_1(\tau)}{\partial\tau}\right]^2+\left[\frac{\partial x_2(\tau)}{\partial\tau}\right]^2. \qquad (1.30)$$

Асимптотическое разложение подынтегрального выражения (1.29) имеет вид

$$\frac{\partial}{\partial\tau}\frac{\partial}{\partial t}H_0^{(2)}(\kappa L)-k^2h(\tau,t)\,H_0^{(2)}(\kappa L_0)\underset{\tau\to t}{=}$$
$$=a(\tau)\frac{1}{\left[\sin\left(\frac{\tau-t}{2}\right)\right]^2}-b(\tau)\ln\left[\sin\left(\frac{\tau-t}{2}\right)\right]+O\left((\tau-t)^2\ln(\tau-t)\right) \qquad (1.31)$$

$$a(\tau)=\frac{i}{2\pi}, \qquad (1.32)$$

$$b(\tau)=-k^2\left(\left(x'\right)^2+\left(y'\right)^2\right)\frac{i}{\pi}. \qquad (1.33)$$

1.2 Методы дискретных особенностей (МДО) применительно к задачам дифракции электромагнитных волн

В случае Е-поляризации дискретная модель ИУ (1.11) получается с помощью замены неизвестной функции $u^{(n)},n=0..N-1,$ и правой части $f^{(n)},n=0..N-1$ на соответствующий интерполяционный полином степени P_n, а также ядра $K^{(m,n)}(\tau,t)$ на его интерполяционный полином степени P_m по переменной τ и полином степени P_n по переменной t:

$$\frac{-2i}{\pi}\int_{-1}^{1}\ln|\tau-t|\frac{u_{P_{m-1}}^{(m)}(t)}{\sqrt{1-t^2}}dt+\sum_{n=0}^{N-1}\int_{-1}^{1}\frac{u_{P_{n-1}}^{(n)}(t)}{\sqrt{1-t^2}}K_{P_{m-1},P_{n-1}}^{(m,n)}(\tau,t)dt=f_{P_{m-1}}^{(m)}(\tau). \qquad (1.34)$$

Используется квадратурная формула интерполяционного типа для

интеграла с логарифмическим ядром [30]:

$$\int_{-1}^{1}\ln|\tau-t|\frac{u_{n-1}(t)}{\sqrt{1-t^2}}dt=-\frac{\pi}{n}\sum_{k=0}^{n-1}u\left(t_k^n\right)\left[\ln 2+2\sum_{p=1}^{n-1}\frac{T_p(\tau)T_p\left(t_k^n\right)}{p}\right], \qquad (1.35)$$

где $u_n(t)$ - интерполяционный полином степени n функции $u(t)$, $T_p(x)$ - полином Чебышева и квадратурной формулой интерполяционного типа для интеграла с гладким ядром

$$\int_{-1}^{1}\frac{u_{n-1}(t)}{\sqrt{1-t^2}}dt=\frac{\pi}{n}\sum_{k=0}^{n-1}u_{n-1}\left(t_k^n\right), \qquad (1.36)$$

где $t_k^n=\cos\left(\pi\frac{2k+1}{2n}\right)$.

Если рассмотрим ИУ(1.34) в точках $t_p^{P_m}, p=0..P_m-1, m=0..N-1$ и воспользоваться квадратурными формулами(1.35, 1.36), получится следующая СЛАУ.

$$AX=B, \qquad (1.37)$$

$$A=\left\{A_{n,m}\right\}_{n,m=0}^{N-1},\ A\in C^{m,m}, m=\sum_{n=0}^{N-1}P_n, \qquad (1.38)$$

где

$$A_{n,m}(i,j)=\begin{cases}\frac{\pi}{P_n}\left(K^{(m,n)}\left(t_j^{(P_m)},t_i^{(P_n)}\right)+\frac{2i}{\pi}\left[\ln 2+2\sum_{p=1}^{P_n-1}\frac{T_p\left(t_j^{(P_m)}\right)T_p\left(t_i^{(P_n)}\right)}{p}\right]\right), if\ m=n\\ \frac{\pi}{P_n}\left(K^{(m,n)}\left(t_j^{(P_m)},t_i^{(P_n)}\right)\right), if\ m\neq n\end{cases},$$

$$B=\left\{B_m\right\}_{m=0}^{N-1}, \qquad (1.39)$$

$$B_m=-\left\{f^{(m)}\left(t_p^{P_m}\right)\right\}_{p=0}^{P_m-1},$$

$$X=\left\{X_n\right\}_{n=0}^{N-1}, \qquad (1.40)$$

$$X_n=\left\{u^{(n)}\left(t_p\right)\right\}_{p=0}^{P_n-1}.$$

Целью компьютерного моделирования дифракции электромагнитных

волн обычно бывает построение диаграммы направленности (для дальней зоны) и полного поля (в точках ближней зоны).

Напряженность электрического поля выражается из (1.6), (1.8), (1.16) по формуле

$$U(x,y)=\sum_{n=0}^{N-1}\left[\int_{-1}^{1}H_0^{(2)}\left(k\sqrt{\left(x^{(n)}(t)-x\right)^2+\left(y^{(n)}(t)-y\right)^2}\right)\frac{u^{(n)}(t)}{\sqrt{1-t^2}}dt\right]. \qquad (1.41)$$

Используя квадратурную формулу (1.36), можно получить выражение для приближенных значений напряженности электрического поля.

$$U(x,y)=\sum_{n=0}^{N-1}\left[\frac{\pi}{P_n}\sum_{p=0}^{P_n-1}H_0^{(2)}\left(k\sqrt{\left(x^{(n)}(t)-x\right)^2+\left(y^{(n)}(t)-y\right)^2}\right)X_{n,p}\right]. \qquad (1.42)$$

Расчетная формула для построения диаграммы направленности рассеянного поля должна вытекать из определения:

$$F(\varphi)=\lim_{r\to\infty}\frac{U}{e^{-ikr}r^{-1/2}}. \qquad (1.43)$$

Далее, как обычно, используется подходящая асимптотическая формула:

$$H_0^{(2)}(z)\underset{z\to\infty}{=}\sqrt{\frac{2}{\pi z}}e^{-iz}. \qquad (1.44)$$

Для получения диаграммы направленности нам понадобится следующий предел:

$$\begin{aligned}&\lim_{r\to\infty}\left[\frac{\sqrt{r}H_0^{(2)}\left(k\sqrt{\left(x(t)-x\right)^2+\left(y(t)-y\right)^2}\right)}{\exp(-ikr)}\right]=\\&=\lim_{r\to\infty}\sqrt{\frac{2}{\pi}}\left[\frac{\sqrt{r}\exp\left(-ik\sqrt{\left(x(t)-x\right)^2+\left(y(t)-y\right)^2}\right)}{\sqrt{k}\left(\left(x(t)-x\right)^2+\left(y(t)-y\right)^2\right)^{1/4}\exp(-ikr)}\right].\end{aligned} \qquad (1.45)$$

В полярной системе координат $x=r\cos\varphi$, $y=r\sin\varphi$, выражение $\sqrt{\left(x(t)-x\right)^2+\left(y(t)-y\right)^2}$ принимает вид:

$$\sqrt{\left(x(t)-x\right)^2+\left(y(t)-y\right)^2}=\sqrt{\left(x(t)-r\cos\varphi\right)^2+\left(y(t)-r\sin\varphi\right)^2}=$$
$$=r\sqrt{\left(\frac{x(t)}{r}-\cos\varphi\right)^2+\left(\frac{y(t)}{r}-\sin\varphi\right)^2}=$$
$$=r\sqrt{\left(\frac{x(t)}{r}\right)^2+\left(\frac{y(t)}{r}\right)^2-\frac{2x(t)\cos\varphi}{r}-\frac{2y(t)\cos\varphi}{r}+\cos^2\varphi+\sin^2\varphi}= \tag{1.46}$$
$$=r\sqrt{1-\frac{2}{r}\left(x(t)\cos\varphi+y(t)\cos\varphi\right)+\frac{y(t)^2+x(t)^2}{r^2}},$$

откуда при $r\to\infty$,

$$\sqrt{\left(x(t)-x\right)^2+\left(y(t)-y\right)^2}\underset{r\to\infty}{=}$$
$$\underset{r\to\infty}{=}r\left(1+\frac{1}{2}\left[-\frac{2}{r}\left(x(t)\cos\varphi+y(t)\cos\varphi\right)\right]+O\left(\frac{1}{r^2}\right)\right)\underset{r\to\infty}{=} \tag{1.47}$$
$$\underset{r\to\infty}{=}r-\left(x(t)\cos\varphi+y(t)\cos\varphi\right)+O\left(\frac{1}{r}\right).$$

Это позволяет выполнить в (1.47) элементарные преобразования и получить результат предельного перехода в виде формулы:

$$\lim_{r\to\infty}\left[\frac{\sqrt{r}H_0^{(2)}\left(k\sqrt{\left(x(t)-x\right)^2+\left(y(t)-y\right)^2}\right)}{\exp\left(-ikr\right)}\right]= \tag{1.48}$$
$$=\sqrt{\frac{2}{\pi k}}\exp\left(ik\cdot\left(x(t)\cos\varphi+y(t)\sin\varphi\right)\right).$$

Используя (1.43), (1.41) и (1.48) получаются выражения для диаграммы направленности

$$F(\varphi)=\sqrt{\frac{2}{\pi k}}\sum_{n=0}^{N-1}\left[\int_{-1}^{1}\exp\left(ik\cdot\left(x(t)\cos\varphi+y(t)\sin\varphi\right)\right)\frac{u^{(n)}(t)}{\sqrt{1-t^2}}dt\right].$$

Воспользовавшись квадратурной формулой для гладкого ядра (1.42) получаем

$$F(\varphi)=\sqrt{\frac{2}{\pi k}}\sum_{n=0}^{N-1}\left[\frac{\pi}{P_n}\sum_{p=0}^{P_n-1}\exp\left(ik\cdot\left(x(t)\cos\varphi+y(t)\sin\varphi\right)\right)X_{n,p}\right].$$

Выпишем формулы, определяющие дискретную модель МДО в задаче дифракции Н-поляризованных волн на проводящих экранах.

Интегро-дифференциальное уравнение (1.29) с учетом (1.31-1.33) переписывается в виде

$$a(\tau)\cdot\int_0^{2\pi}\frac{j(t)}{\sin^2\left(\frac{\tau-t}{2}\right)}dt\ +b(\tau)\cdot\int_0^{2\pi}\ln\left[\sin\left(\frac{\tau-t}{2}\right)\right]j(t)dt+$$
$$+\int_0^{2\pi}K(\tau,t)\,j(t)dt=4h_\tau(\tau)E_\tau^0(\tau), \tag{1.50}$$

где

$$K(\tau,t)=\frac{\partial}{\partial\tau}\frac{\partial}{\partial t}H_0^{(2)}(\kappa L)-k^2h(\tau,t)\,H_0^{(2)}(\kappa L_0)\,u(t)$$
$$-a(\tau)\frac{1}{\left[\sin\left(\frac{\tau-t}{2}\right)\right]^2}+b(\tau)\ln\left[\sin\left(\frac{\tau-t}{2}\right)\right]. \tag{1.51}$$

Подставляя (1.26, 1.27) в (1.20) можно получить выражения для магнитного поля

$$\vec{H}=\left(0,0,H_z(x,y)\right), \tag{1.52}$$

где

$$H_z(x,y)=\frac{1}{4i}\int_0^{2\pi}\left\{\left(y'(t)k\ \frac{x(t)-x}{L_{MN}}-x'(t)\,k\ \frac{y(t)-y}{L_{MN}}\right)H_1^{(2)}(\kappa L_{MN})\right\}j_\tau(t)dt \tag{1.53}$$

Поле в дальней зоне – это зависимость от направления (φ) величины следующего предела

$$F(\varphi)=\lim_{r\to\infty}\frac{\sqrt{r}}{e^{-i(kr-\pi/4)}}H_z(x,y)=F_H(\varphi). \tag{1.54}$$

В дальней зоне он имеет вид

$$F_H(\varphi)=\frac{k}{4i}\sqrt{\frac{2}{\pi\kappa}}\int_0^{2\pi}\left\{-y'(t)\cos\varphi+x'(t)\sin\varphi\right\}ie^{ik\{x(t)\cos\varphi+y(t)\sin\varphi\}}\,j_\tau(t)dt\,, \tag{1.55}$$

$$\phi_k^n = \frac{2\pi k}{2n+1},\ \ k \in Z\ ,\ \phi_{0k}^n = \frac{2\pi\left(k+1/2\right)}{2n+1},\ \ k \in Z\ .$$

Подставив (1.56-1.58) в (1.50) и рассматривая полученное выражение при $\tau = \phi_{0k}^n$ к = 0..2n => получаем 2n + 1 линейных уравнений:

$$\int_0^{2\pi} K(\tau,t)dt \approx \sum_{k=0}^{2n} K(\tau,\phi_k^n)\frac{2\pi}{2n+1}, \tag{1.56}$$

$$\frac{1}{2\pi}\int_0^{2\pi} \frac{f_n(\varphi)}{2\sin^2\dfrac{\varphi-\varphi_0}{2}}d\varphi =$$

$$= \frac{1}{2n+1}\sum_{k=0}^{2n} f_n(\varphi_k^n)\left[\frac{\sin^2\dfrac{n}{2}\left(\varphi_0-\varphi_k^n\right)}{\sin^2\dfrac{1}{2}\left(\varphi_0-\varphi_k^n\right)} - \frac{n\sin\left(n+\dfrac{1}{2}\right)\left(\varphi_0-\varphi_k^n\right)}{\sin\dfrac{1}{2}\left(\varphi_0-\varphi_k^n\right)}\right]. \tag{1.57}$$

$$\frac{1}{\pi}\int_0^{2\pi} \ln\left|\sin\frac{\phi-\phi_0}{2}\right| f(\phi)\,d\phi \approx -\sum_{k=0}^{2n} f(\phi_k^n)\cdot\left[\ln 2 + \sum_{p=1}^{n}\frac{\cos p\left(\phi_0-\phi_k^n\right)}{p}\right]\frac{2}{2n+1}\ , \tag{1.58}$$

$$\frac{2i}{2n+1}\sum_{k=0}^{2n} j_n(\varphi_k^n)\left[\frac{\sin^2\dfrac{n}{2}\left(\varphi_0-\varphi_k^n\right)}{\sin^2\dfrac{1}{2}\left(\varphi_0-\varphi_k^n\right)} - \frac{n\sin\left(n+\dfrac{1}{2}\right)\left(\varphi_0-\varphi_k^n\right)}{\sin\dfrac{1}{2}\left(\varphi_0-\varphi_k^n\right)}\right]$$

$$-b(\tau)\cdot\sum_{k=0}^{2n} j_n(\phi_k^n)\cdot\left[\ln 2 + \sum_{p=1}^{n}\frac{\cos p\left(\phi_0-\phi_k^n\right)}{p}\right]\frac{2}{2n+1} +$$

$$+\sum_{k=0}^{2n} K(\tau,\phi_k^n)\frac{2\pi}{2n+1} = 4h_\tau(\tau)E_\tau^0(\tau).$$

1.3 Пример двумерной задачи дифракции упругих волн

Следуя работам [42, 55, 56], можно построить приближенные численные решения для задачи взаимодействия упругих волн деформации продольного и поперчного типа в неограниченной упругой среде с системой M однородных цилиндров (бесконечных вдоль оси Ox_3), поперечные сечения которых ограничены замкнутыми непересекающимися контурами.

Пусть L – совокупность этих контуров, которая разбивает плоскость Ox_1x_2 на две области: внешнюю D_1 и внутреннюю D_2. (положительное направление обхода на контурах L выбирается так, что при их обходе область D_1 оставалась слева (рис. 1.1).

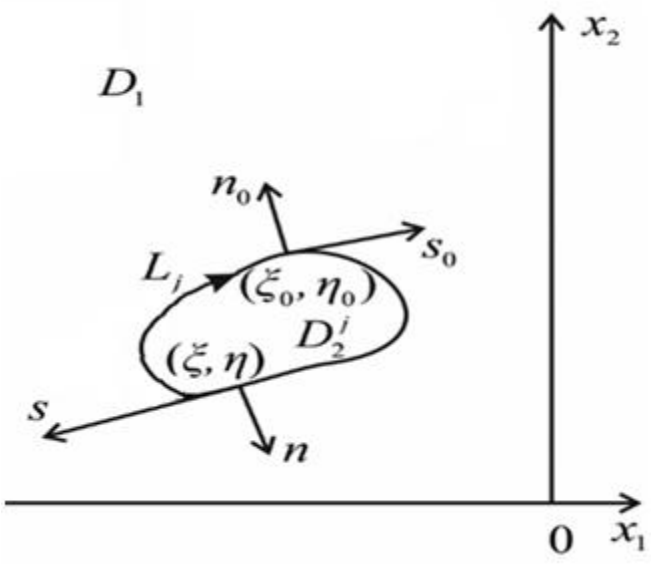


Рис. 1.1 – j-ая неоднородность в неограниченной среде

Цилиндрические включения могут быть полостями, неподвижными или подвижными жесткимим или упругими включениями, ρ_k и λ_k, μ_k – обозначают плотности и коэффициенты Ламе однородного материала (как говорят, матрицы) - индекс $k = 1$ и упругих включений - индекс $k = 2$.

Пусть из бесконечности на цилиндры набегает гармоническая волна расширения-сжатия - P – случай или волна сдвига, SV – случай:

$$U_1^{(0)} = 0,\ U_2^{(0)} = \tau_1 e^{-i\gamma_1^{(1)} x_2},\ \gamma_1^{(1)} = \omega / c_1^{(1)},\ \tau_1 = const, \qquad (1.59)$$

или

$$U_1^{(0)} = \tau_2 e^{-i\gamma_2^{(1)} x_2},\ U_2^{(0)} = 0,\ \gamma_2^{(1)} = \omega / c_2^{(1)},\ \tau_2 = const., \qquad (1.60)$$

где $c_1^{(1)}$ и $c_2^{(1)}$ – скорости P- и SV- волны в матрице.

При взаимодействии приходящих волн со включениями возникают отраженные (и проникающие внутрь упругих включений) упругие волны

только двух типов: продольные и поперечные. Пусть $U_1^{(k)}$ и $U_2^{(k)}$ – амплитуды перемещений отраженного (k=1) и проникающего внутрь упругих включений (k=2) полей соответственно. Общее поле перемещений $\mathbf{U}=\left(U_1, U_2\right)'$ есть $\left(U_1^{(1)}+U_1^{(0)}, U_2^{(1)}+U_2^{(0)}\right)'$ (матрица) и $\left(U_1^{(2)}, U_2^{(2)}\right)'$ внутренность упругих включений. Компоненты вектора $\mathbf{U}$ при временном множителе $\left(e^{-i\omega t}\right)$ удовлетворяют уравнениям:

$$\begin{aligned}(\lambda+2\mu)\frac{\partial^2 U_1}{\partial x_1^2}+\mu\frac{\partial^2 U_1}{\partial x_2^2}+(\lambda+\mu)\frac{\partial^2 U_2}{\partial x_1\partial x_2}+\rho\omega^2 U_1+f_1=0,\\ \mu\frac{\partial^2 U_2}{\partial x_1^2}+(\lambda+2\mu)\frac{\partial^2 U_2}{\partial x_2^2}+(\lambda+\mu)\frac{\partial^2 U_1}{\partial x_1\partial x_2}+\rho\omega^2 U_2+f_2=0,\end{aligned} \qquad (1.61)$$

где $\mathbf{f}=\left(f_1, f_2\right)'=0,$ а на бесконечности рассеянное поле $\mathbf{U}^{(1)}$ удовлетворяет условиям излучения: компоненты поля $U_1^{(1)}$, $U_2^{(1)}$ - расходящиеся волны [56].

Принято добавлять следующие граничные условия [56]. Поперечное сечение каждого цилиндра описывается гладкой замкнутой кривой (уравнения: $\xi=\xi(s),\ \eta=\eta(s)$). Для постановки граничных условий фиксируют точку на её границе $\varsigma_0=\xi_0+i\eta_0$.

На границе раздела двух сред матрица–упругое включение при переходе через L вектора перемещений $\mathbf{U}=\left(U_1, U_2\right)'$ и вектора напряжений $\mathbf{S}=\left(S_1, S_2\right)'$ получают [56]):

$$\left(U_m^{(1)}+U_m^{(0)}\right)_{z\to\zeta_0}=\left(U_m^{(2)}\right)_{z\to\zeta_0},\ \left(S_m^{(1)}+S_m^{(0)}\right)_{z\to\zeta_0}=\left(S_m^{(2)}\right)_{z\to\zeta_0},\ m=1,2. \qquad (1.62)$$

где равенства означают равенство пределов при стремлении точки наблюдения $z=x_1+ix_2\in D_k$ к точке $\varsigma_0=\xi_0+i\eta_0\in L_j$ $(j=\overline{1,M})$ из областей $D_1(k=1)$ и $D_2(k=2)$.

Для *неподвижных включений*

$$\left(U_m^{(1)}+U_m^{(0)}\right)_{z\to\zeta_0}=0,\quad z\in D_1,\ \varsigma_0\in L_j,\ m=1,2,\ j=\overline{1,M}. \qquad (1.63)$$

Для *жестких включений*:

$$\begin{aligned}\left(U_1^{(1)}+U_1^{(0)}\right)_{z\to\zeta_0} &= B_1^j-\omega_j\eta_0,\\ \left(U_2^{(1)}+U_2^{(0)}\right)_{z\to\zeta_0} &= B_2^j+\omega_j\xi_0,\ j=\overline{1,M}.\end{aligned} \tag{1.64}$$

где D_2^j перемещается и крутятся вместе с матрицей; B_1^j и B_2^j – амплитуды поступательного движения, ω_j – жесткого поворота j-го включения.

Для определения постоянных B_1^j, B_2^j и ω_j $\left(j=\overline{1,M}\right)$ в (1.65) используются дополнительные соотношения:

$$\int_{L_j} S_m ds_0 = -q_j B_m^j,\ \ q_j=\omega^2\rho_в S_в^j,\ m=1,2,\ j=\overline{1,M} \tag{1.66}$$

и

$$\int_{L_j}\left(S_1\left(\eta_0-x_{20}\right)-S_2\left(\xi_0-x_{10}\right)\right)ds_0=-\omega^2 J_j\,\omega_j,\quad j=\overline{1,M}, \tag{1.67}$$

где $\rho_в$ – плотность, $S_в^j$ – площадь, J_j – момент инерции включения D_2^j относительно произвольной точки $A(x_{10},x_{20})$.

Для совокупности полостей D_2 с границей L, она свободна от сил, и на L равны нулю компоненты вектора напряжений:

$$\left(S_m^{(1)}+S_m^{(0)}\right)_{z\to\zeta_0}=0,\ z\in D_1,\ \varsigma_0\in L_j,\ m=1,2,\ j=\overline{1,M}. \tag{1.68}$$

1.4 Гауссовы методы решения СЛАУ, используемые в алгоритмах МДО

При использовании МДО требуется решение системы линейных алгебраических уравнений (СЛАУ), которые обладают следующими характеристиками: определитель матрицы гарантированно не равен нулю, система хорошо обусловлена (так как диагональ ярко выражена) и для решения системы не требуется выбор главного элемента. Это приводит нас к использованию метода Гаусса, который состоит из двух этапов (прямого и обратного хода). Согласно [12] для прямого хода требуется порядка $O(N^3)$ операций, в то время как для обратного - $O(N^2)$. В работе [87] показано, что

для использования распределенных вычислений хорошо подходит именно компактная схема Гаусса (в блочном варианте), поскольку у нее более удобная область зависимостей при вычислении очередного элемента.

1.5 Использование параллельных вычислений при реализации МДО

Параллельные вычисления на ПК, в том числе, для методов дискретных токов, какое-то время были доступны лишь в форме образования импровизированных кластеров из компьютеров [82, 87], что позволяет получать ускорение, несколько меньшее числа машин или процессорных ядер. Ускорение, равное и даже несколько большее числа задействованных ядер процессоров, удаётся получить иначе, за счёт особенностей работы с кэшем и при удачном использовании зернистости вычислений [33], на примере умножения матриц (для многих целей в том числе обработка изображений рассмотрение таких операций достаточно). Аналогичные результаты были в свое время получены и для более сложных алгоритмов включая решение линейных систем, однако эти исследования развивались с прицелом на специализированные архитектуры, включая конвейерные и матричные суперЭВМ [13, 62]. Современный подход к исследованию и использованию таких эффектов и организации и использования зернистых вычислений дается в [39, 40]. Долгое время это направление казалось не имеющим отношение к персональным компьютерам. Максимальное возможное для данного типа прикладных задач ускорение в теории можно получить при распараллеливание вычислений на дополнительном специализированном устройстве с большим числом процессорных ядер, наиболее известным примером которого является карточка с поддержкой CUDA [7] или ее аналога OpenCL [97]. Однако, не говоря об экономической стороне дела, приходится иметь в виду, что «При оптимизации или модификации фрагмента программы под определенную аппаратную архитектуру возникает проблема ограничения производительности реализации алгоритма особенностями целевой архитектуры открытой

гетерогенной системы. Эта проблема особенно актуальна при применении специальных вычислительных устройств, таких как графические процессоры, сигнальные процессоры, программируемые логические интегральные схемы, проблемно-ориентированные процессоры» [3].

С этим тесно связаны препятствия, практически возникающие при использовании технологии CUDA в реализации сложных алгоритмов:

- необходимо дополнительное оборудование;
- требуются дополнительные знания - понимание работы CUDA устройства;
- нужно хорошо знать специальный язык программирования для работы с CUDA устройством;
- часто сложность программирования параллельных алгоритмов важных для практики классов превосходит их собственную сложность.

Конкретно по поводу блочных алгоритмов известно [7], что их реализация в открытой гетерогенной вычислительной системе на базе графического процессора осложняется следующими особенностями архитектуры:

1. Высокая параллельность по данным, а не по коду.
2. Структуризация и особенности работы различных типов памяти графического процессора.
3. Малый объем «быстрой» памяти, доступной одному вычислительному потоку.
4. Специфика обработки логических выражений.
5. Специфика обработки условных переходов и циклов.

В связи со сказанным весьма актуален поиск возможностей для ускорения алгоритмов МДО, которые можно найти в базовой архитектуре процессоров современных ПК. Наше внимание привлекли векторные регистры [85]. Первые векторные регистры появились еще в начале 90-х годов, но в то время они могли работать только с целыми числами, их размер был всего 64 бита. Впоследствии с появлением SSE инструкций и

расширением регистров до 128 бит появилась возможность использовать числа с плавающей точкой двойной точности. После SSE появились так называемые AVX инструкции и размер векторных регистров вырос до 256 бит. Самой последней версией набора инструкций работающего с векторными регистрами является AVX512 и, как следует из названия, их размер равен 512 бит. В будущем планируется расширение до 1024, что свидетельствует об актуальности и развитии данной технологии.

Непосредственной целью встраивания векторных регистров в процессоры ПК была, как и в случае с CUDA, ставка на ускорение обработки графической информации. Однако, как и в случае с CUDA, векторные регистры стали использоваться программистами также и при необходимости ускорить вычисления, в том числе с комплексными числами, как это требуется в МДТ. Примеры выполнения комплексного умножения с выигрышем в скорости вычислений для 128-битных SSE инструкций, опубликовано в Интернете [85]. Использование векторных регистров для ускорения обращения разреженных матриц с «маленькими» блоками реализовано и исследовано в [2]. В связи с этим отметим, что так как матрицы МДО полностью заполнены[20], и их разбиение на блоки имеет смысл только в связи с построением параллельных алгоритмов обработки [62, 87].

1.6 Локальность памяти и зернистость вычислений

В отличие от распределения параллельных вычислений на разные компьютеры (или узлы вычислительного кластера), использование нескольких процессорных ядер в составе одного компьютера (узла кластера) сталкивается с возможностью конфликта по разделяемым ресурсам, прежде всего по памяти. Поэтому время выполнения одной и той же программы с помощью многоядерного процессора не всегда будет меньше, чем с помощью одноядерного процессора такой же тактовой частоты. Дело ещё осложняется многоуровневым характером памяти. Чем дольше процессор

работает с кэшем оперативной памяти, не нуждаясь в его обновлении, тем, при прочих равных условиях, быстрее идут вычисления. Этот эффект принято рассматривать как локализацию памяти, и существует общая теория, как им следовало бы управлять (этим вопросам посвящены недавние работы [62, 39]. Но даже при хорошей локализации разные ядра, в принципе, могут задерживать друг друга, используя общий кэш. Необходимо, очевидно, чтобы каждое ядро было занято собственным вычислительным процессом, который хорошо локализуется по памяти. Но априори не ясно, что лучше, чтобы множества данных, с которыми в один и тот же момент работают в кэше разные ядра, не пересекались или же, напротив, были общими? Или требуется какой-то «средний» вариант? Далее, неверен принцип – по одной задаче на одно ядро. Как видно из исследования [33], лучший результат может дать организация в программе значительно большего числа раздельных процессов, чем имеется процессорных ядер.

В реализациях МДО критические по времени выполнения места - формирование матрицы СЛАУ, решение СЛАУ (прежде всего, как следует из предыдущих замечаний, метод Гаусса), расчёт полей по рассчитанным дискретным приближениям к распределениям токов на проводящих поверхностях, визуализация полей.

Оправдана ли вообще забота о локализации памяти при распараллеливании вычислений на разные ядра, и каким методом можно подбирать наилучшие варианты алгоритмических схем распараллеливания, ориентированных на так называемый тайлинг?

Для таких простых вычислений, как перемножение прямоугольных матриц, данная задача ранее была решена на базе тестовой программы, разработанной на языке Ада [33]. Однако, хотя теоретически метод Гаусса и можно свести к последовательному перемножению матриц специального вида, практическое программирование алгоритмов этого метода имеет дело с треугольными частями квадратных матриц и соответственно вложенные суммы имеют переменные пределы. В этом случае не очевидна

целесообразность подбора тайлов на основе квадратных подматриц, и не применимо асинхронное суммирование, в противоположность [33]. Отсюда особенность постановки – требуется следить за тем, чтобы некоторое нарушение регулярности тайлов и дополнительные затраты времени на необходимую синхронизацию раздельно выполняемых процессов не «съели» тот выигрыш от тайлинга, который можно предполагать в алгоритмах, родственных умножению матриц.

1.7 Исследование по энергозатратам программного обеспечения

Перед человечеством стоит острая проблема сохранения природных ресурсов, включая энергосбережение. Информационные технологии являются значительными потребителями электроэнергии (около 3% общего объёма), так что к 2020 г. их вклад в парниковый эффект на планете достигнет 4% [32, 84]. Вопросы экономии энергии при проведении прикладных расчётов требуют, в частности, внедрения более экономичных вычислительных методов. Наряду с повышением экономичности аппаратных и системных программных средств, чем занимается большая индустрия, требуется совершенствование вычислительных методов всех предметных областей. Ответственность за это ложится на специалистов этих областей.

Типичной задачей при конструировании и усовершенствовании многих технических систем является компьютерное моделироване процесса взаимодействия электромагнитного поля с проводящими экранами, которые входят в состав антенных и других устройств. При этом обычно бурутся относительно большие волновые числа, и рассматриваются многие варианты значений параметров конструкций. Это ведёт к большим объёмам вычислительной работы.

Один из наиболее эффективных подходов к компьютерному моделированию дифракционных процессов – численное решение граничных уравнений методами дискретных особенностей (МДО), тем не менее, в силу своей сложности порядка N^3 (при 2-мерных постановках) также требует

длительных вычислений и соответственных трат электроэнергии. Необходимо обновление ПС моделирования. Требуется, чтобы вместо суток расчёты длились часами, а вместо часов - минутами. При этом на те же расчёты не должно уходить больше энергии, чем прежде.

Отсюда интерес к средствам контроля расхода электроэнергии приложениями на фоне решения текущих задач ОС. Такой контроль не нужен, если принять гипотезу Д. Маевского о том, что «зелёное» программное обеспечение (ПО) – то же самое, что и «быстрое» ПО [32, 8 раздел]. Однако, реализация возможностей ускорения, скрытых в архитектуре современных ПК, приводит к более интенсивному энергопотреблению процессора. Обычный энергорасход ПК в разы меньше того максимума, который способны поддерживать его устройства питания. Если одно и то же вычисление удастся вдвое ускорить, а энергопотребление компьютера при этом поднимется втрое, то тезис о равнозначности «зелёный» и быстрый будет опровергнут! Поэтому оценку энергопотребления ПО обойти нельзя.

В работах, посвященных измерению энергопотребления компьютерных программ, отмечается, что соответствующая аппаратура и схемы оценивания должны быть достаточно сложны [32, раздел 6]. Их использование вне специализированных лабораторий проблематично. Программные средства более доступны, но погрешности их измерений оцениваюстся на уровне до 10% [32, С. 250]. Одно из самых известных и используемых из таких средств – Intel Power Gadget, оценивает только энергопотребление процессора.

Выводы по разделу 1

Анализ источников подтверждает актуальность преобразования вычислительных методов класса МДО, созданных для моделирования дифракции, с тем расчётом, чтобы наряду с ограниченным повышением скорости их выполнения на ПК за счёт распараллеливания вычислений на процессорные ядра, подключить с этой целью и другие - скрытые в

архитектуре компьютеров средства. Перспективными для этого на сегодняшний день являются МДО для задач дифракции в 2D постановке. При этом ориентиром успешности новых методов может служить достижение ими ускорения вычислений примерно в 8 раз в сравнении с традиционными методами-прототипами. Существенность такого ускорения объясняется возможностью вдвое увеличить параметр, определяющий точность вычислений, или, сохраняя точность, вдвое увеличить максимальное волновое число при моделировании. Дополнительными вычислительными мощностями, которые можно задействовать в современном ПК являются кеш процессора вместе с оперативной памятью (что имеется ввиду во второй задаче исследования, с.8) и специальные векторные регистры (имеются ввиду в третьей задаче иследования, с.8). Средствами обеспечения должной работы кеша и этих регистров служат, соответственно, управление зернистостью вычислений и их векторизацией (приспособление к выполнению в регистрах). Реализаторы новых методов не должны прилагать усилий, намного больших, чем в случае традиционных методов, что ведет к необходимости согласования, вообще говоря, конкурирующих требований к организации компьютерных вычислений (задача 4, с.8). Так же моделирование дифракции на компьютере новыми методами не должно требовать больше электроэнергии, чем прежде (в связи с чем требуется решать задачу 5, с.8). Таков смысл общей задачи исследования, сформулированной на странице 7, и таким образом уточняются его частные задачи. Отсюда вытекают и все его частные задачи.

РАЗДЕЛ 2
ИСПОЛЬЗОВАНИЕ ЗЕРНИСТОСТИ ДЛЯ УСКОРЕНИЯ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ МЕТОДОВ ДИСКРЕТНЫХ ОСОБЕННОСТЕЙ

На основе концептуальной модели кэширования памяти в ПК и определенных аксиом преимуществ для кэширования различных способов обработки данных в циклах вычислительного метода класса МДО исследовано использование зернистости вычислений, проведена проверка и уточнение выводов методом компьютерного эксперимента. Первые два этапа МДО (общие для них): построение матрицы дискретной модели и решения СЛАУ, - требуют несколько разных подходов к анализу зернистости, но их согласованность обеспечивается тем, что оба требуют выделения в матрице дискретной модели квадратных клеток (размер которых становится параметром метода). Третий этап - непосредственное вычисление параметров дифракции или рассеянных полей - полностью зависит от постановки дифракционной задачи, но приемы придания ему формы, приспособленной к ускоренному выполнению будут комбинацией приемов, разработанных для первых двух этапов. Излагемый материал базируется на работах [45, 49, 52, 65, 66].

2.1 Разработка тайлингового вычислительного метода дискретных особенностей в задаче дифракции электромагнитных волн на экранах

Исследование проводилось при распараллеливании вычислений, с целью усилить его эффект. Ускорение таких вычислений, как МДО, со сложностью по количеству операций N^3 (N - параметр дискретизации), следует считать существенным, когда оно будет 8-кратным, поскольку тогда можно удвоить предельно допустимое для компьютерного моделирования значение указателя дискретизации (улучшая точность моделирования). Для этого, взяв за пример современный ПК с 4-мя процессорными ядрами,

необходимо обеспечить, кроме ускорения выполнения вычислений МДО за счет физического распараллеливания на ядра процессора, дополнительное ускорение, более чем в 2 раза. Резерв составляет преодоления довольно ограниченной (относительно современных технических возможностей) скорости доступа процессора ПК к основной памяти. Процессор современного ПК имеет, как правило 3-уровневый кэш для обмена с основной памятью. Доступ процессора к данным в кэше по крайней мере на порядок быстрее, чем обращение по эти данные непосредственно в памяти. Поэтому скорость той же обработки данных разными методами, теоретически, могла бы меняться почти в 10 раз, и задача ускорить выполнение вычислений МДО, скажем, в 2.5 раза за счет этого эффекта выглядит реальной, хотя и не очевидной и, вообще, не является гарантированной.

Имея ввиду дискретную модель МДО, рассмотрим систему линейных уравнений

$$Ax = f$$

или в подробной записи

$$\begin{pmatrix} a_{11}+a_{12}+...+a_{1n} \\ a_{21}+a_{22}+...+a_{2n} \\ ..... \\ a_{n1}+a_{n2}+...+a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ .... \\ x_n \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ .... \\ f_n \end{pmatrix} \qquad (2.1)$$

Для неё в так называемой компактной схеме метода Гаусса сначала по явным формулам рассчитываются элементы разложения матрицы в произведение левой и правой треугольных матриц (по другой терминологии – нижне и верхнетреугольных). Обозначим ненулевые элементы этих матриц как l_{ij} и u_{ij} соответственно. Тогда, как известно [12], их расчёт можно произвести по схеме:

$$u_{11} = a_{11}, \quad u_{1j} = a_{1j}, \quad l_{j1} = \frac{a_{j1}}{u_{11}}, \quad j = 2,3,...,n, \tag{2.2}$$

$$l_{ii} = 1, \quad u_{ii} = a_{ii} - \sum_{p=1}^{i-1} l_{ip} u_{pi} \quad , \quad i = 2,3,...,n, \tag{2.3}$$

$$u_{ij} = a_{ij} - \sum_{p=1}^{i-1} l_{ip} u_{pj}, \quad l_{ji} = \frac{a_{ji} - \sum_{p=1}^{i-1} l_{jp} u_{pi}}{u_{ii}}, \tag{2.4}$$

$$i = 2,3,...,n, \quad j = i+1, i+2,...,n$$

После этого решение системы (2.1), где $A=LU$, завершается по алгоритму, отражающему процесс последовательного решения двух треугольных систем:

$$L\,y = f \ , \tag{2.5}$$

$$Ux = y, \tag{2.6}$$

что занимает немного (в n раз меньше) времени в сравнении с предвычислением матриц L *и* U. Отметим также, что эти матрицы, как правило, хранятся на месте матрицы A.

Нами было выдвинуто предположение о том, что для вычислительного процесса, который определяется формулами (2.2-2.4), для структуры данных, имеющей форму массива (2.1), можно успешно обеспечить тайлинг (зернистость), предусмотрев отдельные процессы для обработки подматриц вида

$$\begin{pmatrix} a_{i,j} & a_{i,j+1} & \cdots & a_{i,j+m-1} \\ a_{i+1,j} & a_{i+1,j+1} & \cdots & a_{i+1,j+m-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{i+m-1,j} & a_{i+m-1,j+1} & \cdots & a_{i+m-1,j+m-1} \end{pmatrix}, \tag{2.7}$$

$$i = m \cdot k,\ k = 1..\frac{n}{m}, \quad j = m \cdot k,\ k = 1..\frac{n}{m}, \tag{2.8}$$

$$i \neq j \tag{2.9}$$

Впоследствии вычислительные эксперименты подтвердили оправданность этой гипотезы. Действительно, нашлись значения параметров, при которых вычислительный процесс значительно ускоряется, а это в данной ситуации возможно только за счёт того, что при таких значениях параметров обработка данных вида (2.7) действительно становится тайлом.

Однако такой эффект зависит, как отмечено выше, не только от выбора для тайлинга определённых подструктур данных, но и от выбора порядка операций, который обычно можно варьировать в некоторых рамках, не меняя (с точностью до различия вычислительных погрешностей) конечные результаты вычислений. Цель такого варьирования – улучшить локализацию вычислений по памяти. В прозрачных ситуациях на оптимальный выбор варианта может указать результат вычисления так называемых векторов локальности [39, 40]. В любом случае локальность улучшается преобразованиями (перестановкой, разделением операций) циклов, которые позволяют процессорному ядру повторно использовать те же элементы данных на большем числе итераций. Это возможно двумя путями - за счёт улучшения временной и (или) пространственной локальности. Временная локальность означает, что данные повторно используются прежде, чем они перемещаются из памяти с быстрым доступом в память более низкого уровня. Этого можно добиться, сосредотачивая, по возможности, все необходимые операции над порцией данных на итерациях самого внутреннего цикла. Пространственная локальность означает использование в смежных по времени звеньях вычислений данные, расположенные смежно в основной памяти компьютера. Улучшение пространственной локальности повышает вероятность того, что очередная требуемая порция данных

окажется в памяти быстрого доступа, попав туда ранее при загрузке «большого» блока ради другой порции данных.

Несмотря на то, что задача улучшения локальности памяти напоминает задачи оптимизирующего преобразования программ, которые ставятся при компиляции с языков высокого уровня, рассчитывать на автоматическое обеспечение высокой локальности программ штатными компиляторами пока не приходится.

Ручная модификация программ в терминах исходного кода на языке высокого уровня в исследуемой задаче не тривиальна вследствие неизбежного нарушения регулярности тайлов в непрямоугольных структурах. Варианты с относительно лучшей локальностью удалось выявить на основе следующих соображений.

Перестановка индексов суммирования по (i, j, p) в формулах (2.4) порождает множество 3!•3! вариантов, которые мы идентифицируем с помощью пометки подматриц и – по существующей в литературе традиции [33, 39, 62] – упорядоченных троек индексов. Рассмотрим, например, тайл вида (2.7) для U, который должен сохраняться на месте правой верхней треугольной подматрицы (2.1). Если его расчёт производится с сохранением вытекающего из формулы (2.4) порядка перебора индексов, то эта обработка обозначается

$$U\,(i, j, p) \tag{2.10}$$

Если же мы в соответствии с идеями [62, 39] хотим изменить обработку так, чтобы индексы перебирались в другом порядке, например,

$$U\,(i, p, j) \tag{2.11}$$

с сохранением конечного результата, то эта обработка будет соответствовать алгоритму:

$$
\begin{aligned}
&for\ i\ in\ 2..n\ loop\\
&\quad for\ k\ in\ 1..i-1\ loop\\
&\quad\quad for\ j\ in\ k+1..n\ loop\\
&\quad\quad\quad u_{ij} = u_{ij} - l_{ik}u_{kj} \qquad (2.12)\\
&\quad\quad end\ loop;\\
&\quad end\ loo;\\
&end\ loop;
\end{aligned}
$$

Для всех оставшихся 36-2 = 34 вариантов алгоритм обработки тайлов (без изменения конечных результатов) был выписан аналогично, что позволило доказать следующие два утверждения.

Варианты перестановок циклов в нижней треугольной подматрице:

$$L\ (i, p, j),\ L\ (j, p,\ i),\ L\ (p\ ,i,\ j),\ L\ (p,\ j,\ i)\quad , \qquad (2.13)$$

заведомо требуют каждый выполнения более чем на 30 процентов больше операций, чем в каком-то из других вариантов.

Для U менее перспективными по сравнению с другими являются перестановки:

$$U\ (p,\ i,\ j),\ U\ (p,\ j,\ i) \qquad (2.14)$$

Это вытекает из соображений, аналогичных тем, которые в похожих ситуациях приводились в исследованиях [33], по убывания важности:

1. Использование одного и того же значения максимально долгое время.
2. Использование перебора значений по строкам.
3. В случае конфликта между первым и вторым правилом, требуется проведение эксперимента.

Вследствие этих утверждений остаются только 6 перспективных для

экспериментального исследования вариантов:

$$U\,(i, j, p)\; L\;(j, i, p), \tag{2.15}$$

$$U\,(j, i, p)\; L\;(i, j, p), \tag{2.16}$$

$$U\,(i, j, p)\; L\;(i, j, p), \tag{2.17}$$

$$U\,(i, p, j)\; L\;(i, j, p), \tag{2.18}$$

$$U\,(j, i, p)\; L\;(j, i, p), \tag{2.19}$$

$$U\,(i, p, j)\; L\;(j, i, p). \tag{2.20}$$

2.2. Вычислительные эксперименты по уточнению параметров реализации

Перед тем как приступать к проведению основных экспериментов, по которым можно было бы сделать окончательные выводы, необходимы вспомогательные тесты. Эти эксперименты требуются, помимо того, что нужно убедиться в функциональной правильности кода, преимущественно с целью предварительно оценить возможность получения прироста производительности за счёт тайлинга, причём на разных компьютерах. Нами, как вспомогательные, так и основные тесты, проводились на компьютерах Intel atom 330 с 512 kb кэша 2 уровня, Amd Phenom 9550 с 4*512 Kb кэша 2 уровня и 2 Mb кэша 3 уровня и Amd Phenom 8650 с 3*512 Kb кэша 2 уровня и 2 Mb кэша 3 уровня.

В нашем случае предварительное тестирование позволило:

- убедится в том, что само по себе количество выполняемых задач языка Ада свыше числа ядер не влияет на производительность;
- уточнить детали плана основных экспериментов, в частности, наметить

достаточный диапазон варьирования размерностей тайлов.

Следствием первого наблюдения явилась возможность в дальнейшем загружать процессор по максимуму, ориентируясь на простое правило, чтобы Ада задач в наших программах было на одну больше, чем ядер.

Схема основных вычислительных экспериментов такова.

a) Тестирование начинать на одном, практически наиболее доступном для этого процессоре (у нас это был компьютер с Intelatom 330) на матрицах с возрастающими размерностями (мы последовательно генерировали случайные матрицы с 1000x1000, 1731x1731, 2996x2996 и 5186x5186 элементами). Экспериментально оценить зависимость продолжительности вычислений от размера тайла и перестановки циклов. При этом необходимо варьировать перестановки циклов (2.8), проверяя, конечно только те из них, которые рассматриваются как перспективные на основе аналитического исследования локальности (у нас это (2.10)-(2.15)). На выходе первого этапа ожидается, что можно будет выявить наиболее «медленные» (с точки зрения экстраполяционного прогноза) кривые роста времени счёта, как функции размерности. Соответствующие значения параметров (размера тайла и индекса перестановки (2.10)) будут переданы на следующие этапы тестирования для подтверждения или уточнения их оптимальности. Отметим, что в соответствии с результатами предварительного этапа мы рассматривали

тайлы (2.7) с размерностями $m = 20, 40, 80, 150$ (2.21)

(число вариантов в намеченном диапазоне вытекает из оценки ресурса времени).

b) На втором этапе при малом, оставшемся после первого этапа числе параметров (параметры здесь – это размер тайла и перестановка) необходимо убедиться (или опровергнуть), что выявленные на первом этапе закономерности для времени счёта сохраняются на других процессорах. У нас на этом этапе рассматривалось два варианта перестановки циклов

(наилучший на первом этапе и следующий за ним), но все варианты m (2.21) мы из осторожности сохранили. На выходе этапа следует сохранить не более двух перспективных пар параметров и выбрать наиболее производительный компьютер.

c) Попытаться подтвердить оптимальность выбранной пары параметров в рамках избранных дискретных диапазонов их изменения (или, если таких пар две, то окончательно определиться с оптимальной парой, проводя эксперимент с матрицами наибольшей размерности, доступной на наиболее производительном компьютере). В нашем случае треугольное разложение случайных матриц наибольшего из рассмотренных размеров 8977 проводилось на AmdPhenom 9550. Существенно большие размерности были недоступны с точки зрения условия завершения вычислений за 12 часов.

d) Если значительный эффект тайлинга подтвердился, на заключительном этапе целесообразно провести сравнение минимального из наблюдавшихся (при каждом значении размерности) времен счёта с показателем времени для реализаций без тайлинга. Для этих тестов следует подбирать «свой» наиболее производительный компьютер. Мы в этой части эксперимента произвели тесты, использовав процессоры AmdPhenom 9550 и AmdPhenom 8650.

По результатам первого этапа 3.1 выяснилось, что размер тайла не влияет на оптимальную последовательность обработки циклов и оптимальная последовательность для подматрицы L есть (i, j, p), а для U , начиная с размерностей несколько больше 1000, оптимальна последовательность (i, p, j) (для малых же размерностей – (j, i, p)). Поэтому в дальнейшем для L всегда использовалась последовательность (i, j, p), а для U – два варианта (i, p, j) и (j, i, p). Выбор вариантов иллюстрирует рис. 2.1.

Рис. 2.1 – Время выполнения программы в зависимости от перестановки циклов на процессоре atom 330, размер матрицы 2996x2996.

С другой стороны, наилучший размер тайла для каждого значения размерности матрицы свой, причём выбор тестовой матрицы на продолжительность выполнения алгоритма практически не влияет (см. табл. 2.1).

Таблица 2.1.

Наилучший размер тайла в шкале (21) для случайных матриц разного размера, процессор atom 330.

Размер матрицы	1000	1731	2996
Размер тайла	20	40	80
Коэффициент вариации времени счёта при случайном выборе матрицы	0.028	0.013	0.009

На этапе 3.2 эксперименты на всех компьютерах подтвердили, что наилучшими являются схемы обработки:

$$L\ (i,\ j,\ p)\ U\ (i,\ p,\ j)\ . \qquad (2.22)$$

Однако размеры тайлов, оказавшиеся наилучшими, для некоторых компьютеров отличались, как показано в табл. 2.2 (и соответственно на рис. 2.2). Поэтому на следующий этап мы перешли, имея, вообще говоря, по две пары параметров, которые можно было бы рекомендовать как наилучшие (например, для размерности 2996 это 80 и 40).

Таблица 2.2.

Наилучший размер тайла в шкале (2.21) для матриц разной размерности (обработка тайлов по схеме (2.22), данные на примере процессора Phenom 8650).

Размер матрицы	1000	1731	2996	5186	8977
Размер тайла	20	20	40	80	150
Вариация времени счёта при варьировании размерностей тайлов (2.21)	1.54	0.75	0.15	0.05	0.10

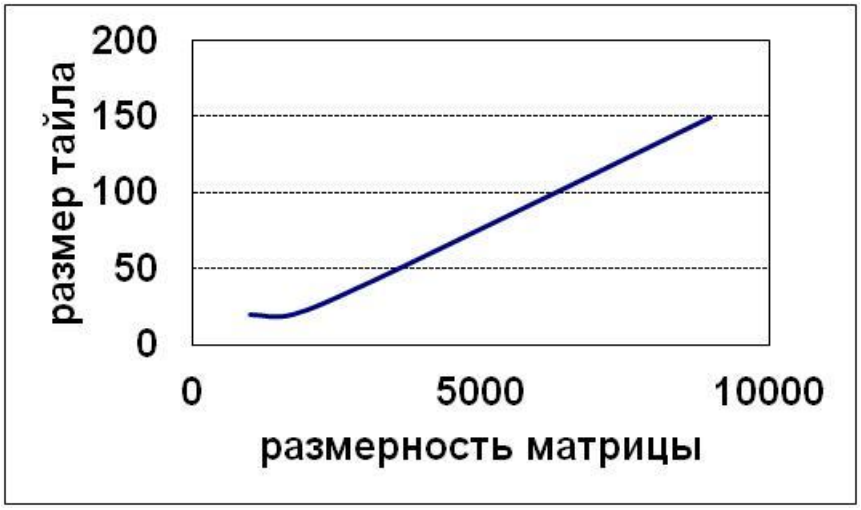


Рис. 2.2 – Наглядная зависимость оптимального размера тайла от размера матрицы согласно данным табл. 2.2.

Для третьего этапа был выбран процессор AmdPhenom 9550. Хотя его производительность на ядро меньше по сравнению с AmdPhenom 8650, но из-за большего количества ядер суммарная производительность у него выше. Результаты показаны в табл. 2.3. Отметим, что хотя для размерности 2996 в

соответствии с нашей схемой экспериментов нужно было выбрать в качестве рекомендуемого варианта размер тайла 40, время вычислений для наиболее мощного выбранного процессора практически не изменится и при выборе m=20.

На основе табл. 2.1-2.3 можно сделать вывод о том, что при увеличении размера матрицы (2.1) начиная с некоторого значения, увеличивается и размер рекомендуемого тайла, но чем больше кэш памяти и количества ядер у процессора, тем более медленным становится это увеличение.

Таблица 2.3.

Наилучший размер тайла в шкале (2.21) для случайных матриц разной размерност в экспериментах на компьютере с процессором Phenom 9550.

Размер матрицы	1000	1731	2996	5186	8977
Размер тайла	20	20	40	40	80
Относительное время счёта	1.00	6.02	36.3	207	1300

На заключительном этапе 3.4 проведилось тестирование многопоточных программ без тайлинга и с тайлингом, а также однопоточной программы (без тайлинга), результаты показаны на рис. 2.3.

2.3. Использование локализации памяти для процесса вычисления элементов матрицы

При дискретизации системы интегральных уравнений, большая часть времени тратится на заполнение матрицы. Согласно (2.23) заполняемая матрица состоит из блоков $A_{n,m}$, элементы которых вычисляются, используя контуры C_n и C_m. Так как на C_n мы всегда используем одни и те же точки $C_n^i \left(i = 0..P_n - 1\right)$, то логично вычислить их заранее.

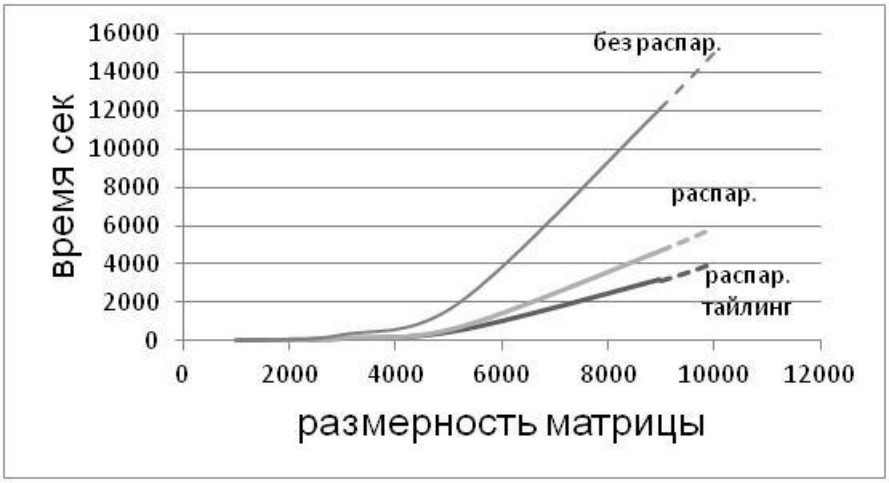


Рис. 2.3 – Время расчета однопоточной программы и распараллеленных программ с тайлингом и без на компьютере с процессором Phenom9550

Каждый элемент матрицы вычисляется независимо. При этом используется относительно небольшое количество данных C_n^i.

Расчет элементов матрицы происходит в 4 вложенных циклах

```
for (int i = 0; i < N; i++)
  for (int j = 0; j < N; j++)
    for (int ii = 0; ii < c[i].size(); ii ++)
      for (int jj = 0; jj < c[j].size(); jj++)
```

где N – количество контуров, c[k].size() – количество точек на k-том контуре.

Первый цикл отвечает за расчет полос состоящих из блоков $A_{n,m}$, второй – за расчет значений самих блоков $A_{n,m}$, третий – за расчет строк состоящих уже из элементов матрицы (блока) и четвертый – уже за расчет самих элементов. В большинстве реальных задачах количество контуров варьируется от 2 до 10, поэтому распараллеливание внутри первого цикла нецелесообразно, так как количество полос может часто окажется меньше количества ядер в системе. Распараллеливание внутри четвертого цикла, также не имеет смысла рассматривать, так как если задание для каждого

потока не будет определено заранее (что часто бывает), то появится достаточно много накладных расходов на динамическое распределение нагрузки между потоками. Распараллеливание модификации по столбцам внутри блока (перестановка 3 и 4 цикла) мы не рассматриваем, так как это заведомо медленнее. Остается всего два вида распараллеливания:

1) по блокам $A_{n,m}$ (внутри второго цикла);

2) по строкам внутри блока (внутри третьего цикла).

Из всего перечисленного выше видно, что скорость вычисления будет зависеть от локальности памяти для элементов C_n^i. Можно выдвинуть гипотезу о том, что при распараллеливании по строкам внутри блока мы должны получить лучшую скорость счета по сравнению с распараллеливанием по блокам. Для проверки этой гипотезы эксперименты проводились при N=2, N=5 и N=10, количество точек дискретизации для каждого контура было P_n = 100, P_n=300, P_n=800.

Таблица 2.4.

Результаты экспериментов N=2, 5, 10

Pn	KB	KS	t, c., N=2	t, c., N=5	t, c., N=10
100	1	1	0,089	0,252	0,595
100	1	4	0,058	0,138	0,306
100	4	1	0,053	0,148	0,272
100	4	4	0,051	0,124	0,268
300	1	1	2,166	5,662	11,77
300	1	4	0,983	2,651	5,164
300	4	1	1,142	2,935	5,395
300	4	4	0,906	2,354	5,175
800	1	1	38,166	98,016	204,88
800	1	4	16,612	41,895	86,71
800	4	1	21,567	52,665	91,558
800	4	4	16,435	41,786	87,44

Обозначим потоки внутри блока KB, а потоки по строкам внутри блока – KS.

Каждый эксперимент состоял из 4 тестов:

1. без распараллеливания;
2. распараллеливание по строкам внутри блока;
3. распараллеливание по блокам;
4. распараллеливание одновременно двумя способами.

Эксперименты проводились на процессоре core i5-2430M (2 cores with hyper threading), компилятор Visual Studio 2013.

Результаты экспериментов подтвердили нашу гипотезу о том, что распараллеливание по строкам внутри блоков быстрее распараллеливания по блокам, причем разница в скорости может превышать 20%.

Основной эффект достигнут, однако имеется некоторый резерв улучшения зернистости, который имеет смысл изучить, поскольку решение СЛАУ – необходимый атрибут всех известных МДО, и соответствующую программу можно импортировать во все реализации. Речь идёт о возможном улучшении пространственной локальности.

Следуя в целом подходу к распараллеливанию решения СЛАУ из подраздела 2.1 (компактная схема Гаусса), дополнительно рассмотрим перестановки циклов для нижнетреугольной матрицы, которые прежде на определенных основаниях априори считались бесперспективными для оптимизации. Речь идет о перестановках в следующем фрагменте алгоритма:

```
for (inti = 0; i<size; i++) {
//для верхнетреугольной матрицы
for (intj = i; j<size; j++)
for (int k = 0; k < i; k++)
    matrix[i][j] -= matrix [i][k] * matrix [k][j];
//для нижнетреугольной матрицы
for (int j = i + 1; j<size; j++) {
MatrixElementType sum = 0;
```

```
for (int k = 0; k < i; k++)
     sum += matrix [j][k] * matrix [k][i];
   matrix [j][i] = (matrix [j][i] - sum) / matrix[i][i];
}
}
```

Здесь вложенность циклов соответствует порядку индексов (i, j, k).

Перспективным преобразованием этого фрагмента для нижнетреугольной матрицы является такое, которое соответствует порядку циклов (j, k, i). Мы заметили это по прямой аналогии с найденным ранее в (2.2) лучшим (в численных экспериментах) порядком (i, k, j) для верхнетреугольной матрицы.

В этом перспективном варианте необходимо преодолеть недостаток, связанный с увеличением числа арифметических операций. Этого удалось достичь благодаря дополнительному разделению нижнетреугольной матрицы на небольшие подматрицы (их размер подобран в численных экспериментах) и разбиению вычислений на два этапа. Первый этап полностью аналогичен нахождению верхнетреугольной матрицы, а следовательно лучшим порядком циклов по индексам станет (j, k, i), в примере ниже (ij, ik, ii). Второй этап позволил сэкономить вычисления за счет переноса в него одной общей операции деления:

```
for (int i = 0; i < bn; i++){
….
for (int j = i + 1; j < bn; j++) {
  for (int k = 0; k < i; k++) {
   for (int ij=lBorder(j), rbj=rBorder(j); ij<rbj; ij++)
// первыйэтап
for (int ik=lBorder(k), rbk=rBorder(k); ik<rbk; ik++)
     for (int ii = lBorder(i), rbi = rBorder(i); ii < rbi; ii++)
      luMatrix[ij][ii] -= luMatrix[ij][ik] * luMatrix[ik][ii];
}
```

```
for (int ii = lBorder(i), rbi = rBorder(i); ii < rbi; ii++){
// второйэтап
for (int ik = lBorder(i); ik < ii; ik++)
    for (int ij = lBorder(j), rbj = rBorder(j); ij < rbj; ij++)
     luMatrix[ij][ii] -= luMatrix[ij][ik] * luMatrix[ik][ii];
    for (int ij = lBorder(j), rbj = rBorder(j); ij < rbj; ij++)
     luMatrix[ij][ii] /= luMatrix[ii][ii];
}
 }
}
```

где bn – количество подматриц в строке или столбце, индексы i, j, k отвечают за перебор подматриц, lBorder(p) и rBorder(p) левая и правая граница подматрицы.

Покажем оптимальность в численных экспериментах. Таблица сравнения обычной реализации, реализации с улучшенной локальностью памяти и параллельной приведена в таблице ниже. Реализация с улучшенной локальностью памяти по своей структуре практически полностью соответствует параллельной. Эксперименты проводились на процессоре core i5-2430M (2 cores with hyper threading), компилятор Visual Studio 2013.

Таблица 2.5.

Результаты экспериментов решения СЛАУ при разной реализации

Размерность матрицы	1000	1731	2996	5186	8978
Обычная реализация	1.325	10.39	68.2	395.8	2172
Реализация с улучшенной локальностью памяти	0.409	2.141	11.4	59.3	310.3
Параллельная реализация	0.215	1.123	5.88	30.8	161.3

Как видно из табл. 2.5 при переходе от обычной реализации к реализации с улучшенной локальностью памяти за счет обработки матрицы

по подматрицам и перестановки циклов наблюдается прирост производительности порядка 3-7 раз, причем при увеличении размерности матрицы, разница в производительности увеличивается, коэффициент распараллеливания, как и раньше, остается равен количеству ядер.

Эффект сохраняется при использовании различных компиляторов, тестирование проводилось на gcc4.7, Visual Studio 2012, Visual Studio 2013 и mingww-64(gcc4.8.2 posix threading model).

Сравнение скорости счета распараллеленной реализации с обычной.

Определим стандартное минимальное численное решение (СМЧР) дифракционной задачи как нахождение «дискретных токов» из дискретной модели уравнения (2.19) (решение СЛАУ) и по нему – поля в дальней зоне, т.е. диаграммы направленности по напряженности поля с шагом 1˚, что обычно достаточно для демонстрации основных эффектов. СМЧР делает осмысленным сравнение счета на основе разных методов. Результаты сравнения программной реализации S традиционного метода дискретных токов с его тайлинговой параллельной реализацией M (по методу настоящей работы) приведены в табл. 2.6. При этом S – однопоточная программа, а M – многопоточная с квазиоптимальным размером зерна вычисления.

Таблица 2.6.

Время T получения СМЧР (N = 1)

Pn	Заполнение матрицы		решение СЛАУ		время	
	S	M	S	M	S	M
100	0,206	0,038	0,001	0,001	0,211	0,043
200	1,53	0,238	0,014	0,009	1,562	0,255
400	10,16	1,861	0,119	0,061	12,14	1,938
800	103,7	13,44	1,092	0,24	104,9	13,72
1200	346,4	43,42	4,152	0,693	350,6	44,16
1600	810,7	108	10,67	1,482	821,4	109,6
2000	1559	199,1	23,78	3,508	1582	202,7

Регрессия T на N (табл. 2.6) подтверждает сложность СМЧР МДТ как $T=C\,Pn^{\gamma}$ ($C=2{,}07\cdot10^{-7}$; $\gamma=2{,}99$), но тайлинг снижает C, и тем сильнее, чем выше размерность (Pn). Поэтому теперь регрессия даёт $C=7{,}96\cdot10^{-8}$; $\gamma=2{,}84$.

Тесты проводились на процессоре Intel Xeon Processor E5645 (12M Cache, 2.40 GHz, 5.86 GT/s Intel QPI hyper threading) 6 ядер.

Таблица 2.7.

Время получения СМЧР (N = 2)

Pn	Заполнение матрицы		решение СЛАУ		время	
	S	M	S	M	S	M
100	0,404	0,088	0,014	0,009	0,426	0,105
200	3,196	0,495	0,121	0,048	3,334	0,56
400	24,66	3,491	1,038	0,238	26,045	4,079
800	209,8	27,54	11,07	1,494	221,5	29,7
1200	696,2	88,14	42,85	4,828	739,14	93,07
1600	1634	205,3	113,2	11,29	1746,9	216,8
2000	3122	403,3	250,1	21,9	3372,6	425,3

Таблица 2.8.

Время получения СМЧР (N = 5)

Pn	Заполнение матрицы		решение СЛАУ		время	
	S	M	S	M	S	M
100	1,132	0,175	0,236	0,078	1,389	0,274
200	8,144	1,141	2,249	0,404	10,43	1,588
400	67,18	8,489	25,11	2,856	92,38	11,432
800	521,1	67,54	256,5	21,88	777,8	89,591
1200	1739	216,1	1038	73,83	2778	290,16
1600	4064	509,4	2513	174,1	6577	683,85
2000	7927	1020	5032	340,8	12960	1361

Из приведенных таблиц видно, что коэффициент распараллеливания тем выше, чем больше размерность матрицы, с которой мы работаем и в основном больше или равен количеству ядер в системе.

Таблица 2.9.

Время получения СМЧР (N = 10)

Pn	Заполнение матрицы		решение СЛАУ		время	
	S	M	S	M	S	M
100	2,47	0,351	2,925	0,429	5,436	0,821
200	17,75	2,381	24,31	2,92	42,13	5,39
400	134,7	17,5	259,7	22,2	394,6	39,86
800	1075	134,2	2648	177,8	3723	312,4
1200	3573	447,9	8982	596,9	12556	1045
1600	8285	1044	21197	1419	29483	2464
2000	16009	2101	41493	2722	57503	4824

Таблица 2.10.

Зависимость времени счета от количества контуров для параллельной реализации

Pn	N			
	1	2	5	10
100	0,043	0,105	0,274	0,821
200	0,255	0,56	1,588	5,39
400	1,938	4,079	11,432	39,86
800	13,72	29,7	89,591	312,4
1200	44,16	93,07	290,16	1045
1600	109,6	216,8	683,85	2464
2000	202,7	425,3	1361	4824

Удобство модельного примера, в котором одинаковые ленточные экраны выстраиваются в три ряда, состоит в том, что конфигурация примерно сохраняется при довольно широком варьировании числа лент в ряду. Это даёт основание экспериментировать, зависит ли и как скорость реализации разработанного тайлингового метода МДО от числа экранов.

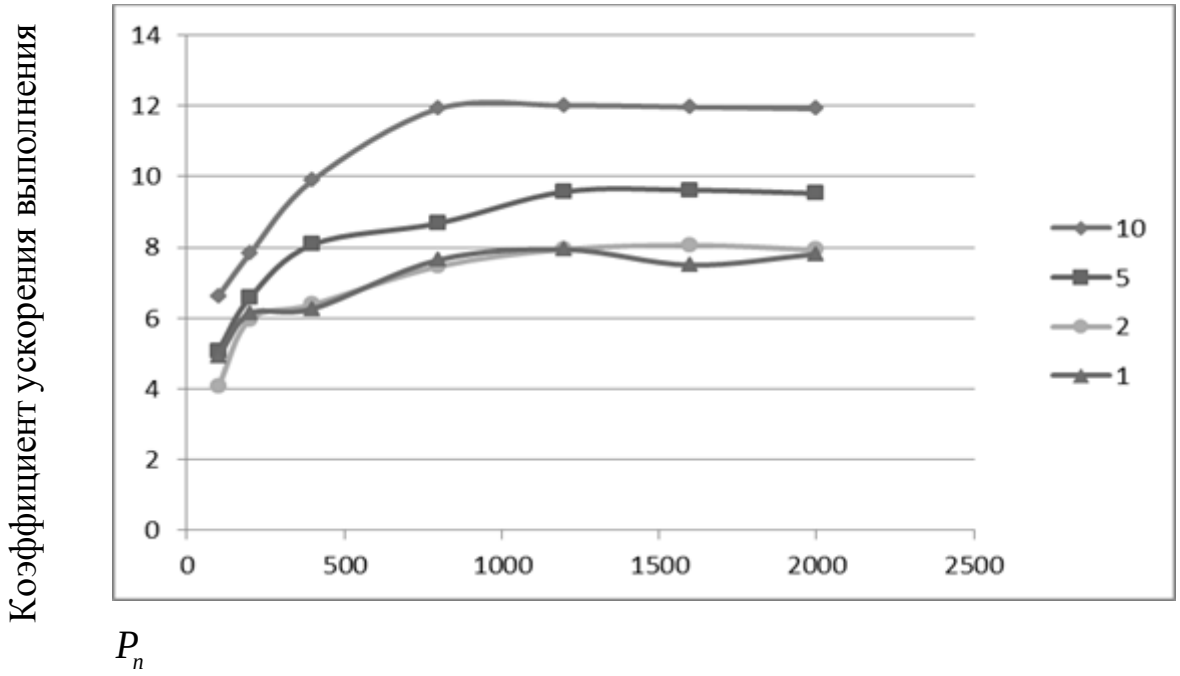


Рис. 2.4 – Коэффициент ускорения для разного количества контуров

2.4 Тайлинговый метод для распределенных систем

Для распараллеливания любой программы между несколькими узлами (компьютерами), основной проблемой является выбор зерна вычисления, но в данном случае зерна вычислений уже выбраны, поэтому специфической задачей является добавление использования процессорных ядер принадлежащих разным узлам, т. е. при нарушении модели вычислений с общей памятью (при компьютерной реализации это приводит к использованию сокетного обмена). При этом появляются такие проблемы:

- нельзя полностью разделить обработку L и U матриц, так как при вычислении элементов лежащих на главной диагонали у обеих матриц требуется синхронизация, а при использовании сети это достаточно

большие задержки;

- нельзя заранее распределить вычислительную работу между узлами, так как отсутствует информация о продолжительности вычислений;
- предыдущая проблема усугубляется в случае разной производительности разных узлов;
- вычислительные процессы на разных узлах не имеют доступ к одному монитору для синхронизации;
- усложняется синхронизация между параллельными процессами, так как у узлов отсутствует не только общая память, но и общий кэш;
- передача данных между узлами относительно медленна и чтобы не было простоя вычислений, приходится наперед обеспечить узлы запасом вычислительной работы

Эти проблемы мы решаем следующим образом:

- выделен и оформлен новый вид вычислений для обработки центральных блоков матрицы (находящихся на пересечении L и U подматриц), это позволило уменьшить количество синхронизаций;
- один из узлов считается главным (сервером), он должен рассчитывать центральные блоки матриц и раздавать задания другим узлам;
- вычислительные задания раздаются по запросу, так, что по окончании выполнения текущего поток делает запрос на сервер и получает очередное задание, это одна из мер по борьбе с простоями узлов кластера, другая мера это избыточность задач, распределенных на один узел;
- поддерживать два типа синхронизации (для сервера и для клиентов);
- разнести процессы отправки и получения данных.

Такой подход при реализации позволяет одноразово открыть соединение и пользоваться им до окончания расчетов. Для того чтобы отправить данные их просто нужно поместить в очередь отправки и указать тип этих данных. Задача для получения данных сама распределяет их другим

задачам, для которых они предназначены.

Успешность этих решений поясним на примере параллельного расчета, на кластере из персональных компьютеров для подматриц L и U матрицы дискретной модели. Этот расчет идет по следующей схеме.

1. Поток вызывает на мониторе клиента функцию получения нового задания и блокируется.
2. Разблокируется задача отправки данных на клиенте и делает запрос на сервер о получении нового задания.
3. Задача получения данных на сервере принимает запрос и вызывает на мониторе сервера функцию получения нового задания.
4. Разблокируется задача отправки данных на сервере и передает новое задание на клиент.
5. Задача получения данных на клиенте помещает полученное задание в монитор клиента.
6. Поток разблокируется и начинает выполнять задание.

Подобная схема позволила каждому потоку для расчета матриц L и U считать, что он выполняется на сервере.

Метод организации вычислительных экспериментов на кластере

Тесты проводились на двух 4-х ядерных компьютерах Amd Phenom 9550 с 4*512 Kb кэша 2 уровня и 2 Mb кэша 3 уровня, связанных между собой через свитч, скорость сети 1 Гбит.

Время расчета высчитывалось по формуле:

Для однопоточного алгоритма и параллельного с тайлингом, время расчета = время окончания расчета - время начала расчета.

Для параллельного алгоритма с тайлингом на двух компьютерах, время окончания расчета - время формирования исходных данных, то есть сюда включено время на пересылку изначальной матрицы между компьютерами.

Результаты тестов под linux приведены в следующей таблице

Таблица 2.11.

Сравнение продолжительности вычислений (в сек.) для программ с

разной организацией (размер тайла 20).

Организация программы	Размерность системы			
	1000	1731	2996	5186
Однопоточная	11,3	59,27	341	1964
Распараллеленная с тайлингом	2,44	14,73	89	508
Распараллеленная между двумя компьютерами с тайлингом	2,05	7,96	46	255

Как видно из таблицы приведенной выше при использовании тайлинга прирост производительности при использовании всех четырех ядер процессора составляет более 380 процентов, а при использовании второго компьютера (суммарное количество ядер процессоров 8) прирост производительности составляет свыше 740 процентов. Прирост производительности практически не наблюдается в первом случае при расчете матрицы размером 1000, это можно объяснить тем, что время нужное для установления соединения и передачи исходных данных сравнимо со временем расчета.

Провести тест под операционной системой Windows оказалось неактуально в связи с тем, что встроенная поддержка сетевых технологий в GNAT под эту ОС не оптимизирована (скорость передачи данных на уровне 4Мбит/с) и прирост быстродействия от добавления второго узла измерялся несколькими процентами. Эксперимент проводился на платформе linux, дистрибутив Pelican.

2.5. Распространение тайлингового метода на другие МДО

Схема вычислений по любому МДО в задачах дифракции содержит 3 этапа: формирование расширенной матрицы дискретной модели, решение СЛАУ с этой матрицей, преобразование этого решения в приближения к искомым характеристикам дифракционного процесса. На втором этапе

- заимствовать из схемы МДТ (и его реализации) подсистему решения

СЛАУ Gauss без изменений, если нет необходимости отказаться от метода Гаусса (о чём никогда не сообщалось, хотя известно о фактах привлечения в МДО также других прямых и итерационных методов).

Следует использовать первый этап МДТ как шаблон аналогичных решений первого этапа других МДО, когда, в общем случае, имеется не дискретная модель системы однотипных, но разных ядрам и роду уравнений, возможно, с дополнительными условиями на решение [23, 55]. Это требует:

- самостоятельно переосмыслить структуру данных исходного МДО для клеточного представления матрицы дискретной модели;
- выбрать аналогично подразделу 2.3 «быструю» схему вычисления элементов этой матрицы (из числа математически эквивалентных схем).

3-й этап при минимальных требованиях, судя по опыту рассмотренных СМЧР, может занимать порядка процента всего времени счёта. Поэтому

- выяснить, насколько актуально усиливать распараллеливание вычислений третьего этапа заботами о зернистости этих вычислений;
- если – да, то действовать, как на 1-м этапе, возможно, и как в 2.2.

Наконец,

- обеспечить эффект локальности памяти при многопоточной (также однопоточной) реализации данного МДО согласованным (между этапами) выбором размера зерна и подтвердить это экспериментально.

2.6 Разработанные методы компьютерного моделирования взаимодействия электромагнитных волн с металлическими экранами

Дифракционные процессы наиболее часто возникают и требуют своего компьютерного моделирования при взаимодействии с препятствиями звуковых, электромагнитных и упругих волн. Мы подробнее останавливаемся на взаимодействии с препятствиями (ограничившись идеально проводящими экранами) электромагнитных волн, поскольку это важнейший и типичный случай. Отметим, что скалярная дифракция (звук)

существенно проще за счёт отсутствия эффекта поляризации волн, а дифракция волн деформации-сдвига технически несколько сложнее электродинамического случая [55], и в дальнейшем мы при тестировании методов диссертации мы обращаемся к такому случаю, хотя и не самому общему.

2.6.1. Случай E-поляризации

Выше мы использовали именно случай E-поляризации в качестве полигона проверки и примера демонстрации эффекта от тайлинг-ориентированной перестройки МДТ. Однако осталась вне рассмотрения важная сторона такой перестройки, которую нудно рассматривать как специальный вид устойчивости МДО. Не происходит ли так, что с ростом размерности задачи, а разрабатываемые методы нацелены именно на применение в сложных задачах, в которых ускорение выполнения расчётов весьма актуально, что будет снижаться точность компьютерного решения СМЧР за счёт накопления вычислительных погрешностей? Это вопрос не праздный – хорошо известно на примерах, что отклонение от вычислительной схемы конкретного метода, зарекомендовавшего свою точность, приводит к потере этой точности.

Оказалось, однако, что числовые значения, полученные по тайлинговому параллельному методу и по традиционному МДТ совпадают (для плотности тока на экранах до 8-го знака мантиссы), и практически совпадают (до 5-го знака) с доступными результатами, полученными с помощью реализации традиционного МДТ другими авторами по другому вариантом метода (сравнивали с результатами работы системы EDEM2, которая, правда, в доступном режиме ограничена в допустимом количестве дискретных особенностей). С ростом размерности вычислительная устойчивость сохранялась.

Обобщим результаты экспериментов по ускорению (табл. 2.9). Ускорение в степени, превышающей число доступных процессорных ядер,

имеет место для всех задач компьютерного моделирования дифракционного процесса с имеющим смысл параметром дискретизации (он должен, как известно [70, 37], обеспечивать не менее 5 дискретных особенностей на длину поперечного контура экрана, равную длине волны). Далее, имеем такую закономерность: *существенное* ускорение, которое должно быть не менее чем в 8 раз (приблизительно - это не чёткий рубеж) за счёт распараллеливания с тайлингом имеет место для компьютеров с 4 или более ядрами при размерности дискретной модели $2 \cdot 10^3$ и более. Свою роль играет и число проводящих экранов. Если длины контуров их сечений примерно равны, то чем их больше, тем сильнее ускорение.

Таблица 2.12.

Типичные значения коэффициентов ускорения компьютерного решения СМЧР

N - параметр дискрет.	**t1** - время вычисления для традиционного МДТ			**t2** - время вычисления для тайлингового. метода			**t1/t2** - ускорение вычислений		
	2 ядра	4 ядра	6 ядер	2 ядра	4 ядра	6 ядер	2 ядра	4 ядра	6 ядер
20x50	36.45	6.81	5.36	10.6	1.02	0.82	3.44	6.68	6.54
20x100	334	72.5	42.13	87.4	8.42	5.39	3.82	**8.61**	**7.68**
20x200	3109	684	394.6	785	67.9	39.86	3.96	**10.07**	**9.9**
20x400		6712	3723		547	312.4		**12.27**	**11.92**

2.6.2. Случай H-поляризации

Существует разница в ускорении для E- и H-случаев, в которых используются предельные уравнения с различными особенностями в ядрах, но на скорость выполнения влияет не порядок особенности в ядре, а количество вычислений элементарных и цилиндрических функций в расчете на один элемент матрицы дискретной модели. Существенное ускорение достигается на 4-х ядрах в H-случае в упомянутом модельном примере при 100 дискретных особенностях на каждом из 20 экранов.

2.7 Ускорение компьютерного моделирования дифракции упругих волн

Рассмотрены также результаты переноса решений тайлингового параллельного метода класса МДТ на задачи взаимодействия упругих волн антиплоской деформации на эллиптических полостях или включениях. В связи с этим примером указано преимущества первичной реализации тайлингового параллельного метода по сравнению с модификацией существующей реализации традиционного МДО. В табл. 2.13 приведены средние по разным видам задач ускорения в задачах рассеяния упругих волн в случаях одинокой полости, упругого и твердого включений, линейно-периодической системы включений, $\kappa = 2\pi$).

Таблица 2.13.

Сравнение традиционной реализаций МДО и реализации тайлингового метода, полученной последовательным абгрейдом существующей ПС в задачах дифракции упругих волн

Стадия абгрейда:	По сравнению с исходной программой	Усилия:
обновление решения СЛАР	ускорение в 1.5 раза	нет вовсе
+ обновление построения матрицы дискретной модели	ускорение в 2.1 раза	незначительные – действия по шаблону
+ обновление расчета перемещений и напряжений	ускорение в 2.4 раза	нужно творчески комбинировать известные операции

Выводы по разделу 2

Впервые разработан тайлинговый параллельный вычислительный метод 2D моделирования дифракции электромагнитных волн на проводящих экранах. Разработанный метод относится к группе МДТ, однако отличается схемой вычислений от известных методов, считая параллельные. Новая схема вычислений позволяет улучшить локальность памяти при реализации в

мультиядерной системе одного или нескольких ПК, существенно уменьшая время расчета матрицы, решение СЛАУ дискретной модели граничных уравнений и диаграмм направленности. Так же данная схема предназначена служить шаблоном аналогичных решений для других МДО. Как показывает пример с дифракцией E-поляризованных электромагнитных волн на системе 20 ленточных металлических экранов, разработанный метод ускоряет приближенное вычисление плотности токов на экранах и диаграммы направленности при 100 дискретных особенностях на экране почти в 4 раза на 2-ядерном ПК и примерно в 8 раз на 4-х и 6-ти ядерном по сравнению с традиционным методом.

Так же стоит отметить, что, хотя распараллеливание вычислительного процесса моделирования дифракционных процессов МДО за счёт многопоточности и даёт значительную экономию времени на многоядерном процессоре (примерно в количество ядер согласно рис. 2.1-2.4), она еще в несколько раз увеличится, если задействовать тайлинг. В практике решения больших задач, требующих суток времени на счёт (размерность около 10^4), а также при решении большого числа задач такая дополнительная экономия, безусловно, существенна.

Другой важный вывод относится к методу исследования. В настоящее время вследствие того, что многоядерные процессоры устанавливаются на персональные компьютеры, прикладным программистам нередко приходится сталкиваться с задачами распараллеливания вычислений, относящихся к той или иной прикладной области. Например, это касается тех исследователей, которые работают над проблемой реализации вычислений МДО в области дифракционных задач акустики и электродинамики. Популярный подход состоит в том, чтобы использовать дополнительные инструментальные средства (компиляторы, библиотеки), создаваемые для инициации и управления несколькими (иногда многими) вычислительными процессами на компьютере. При этом код первоначально однопоточной программы, реализующей заданный алгоритм, анализируется на наличие «узких мест»,

которые подлежат «расшивке» за счёт параллельности. Окончательная параллельная версия программы создаётся фактически вручную за счёт искусства и опыта разработчика ad hoc. Примером может служить недавняя работа [29] тоже посвященная численной реализации МДО, но в области фильтрационной гидродинамики с использованием классической схемы метода Гаусса и известной библиотеки OpenMP. В ней на 4-ядерном процессоре в диапазоне размерностей до $0.4 \cdot 10^2$ достигнуто ускорение в решении СЛАУ не меньше, чем примерно в 3.5 раза (со снижением этого показателя при увеличении размерности). Мы добились намного лучшего результата для размерностей от 10^3 до 10^4, применив свой вычислительный метод, и даём рекомендацию об устройстве программы с единственным, в конечном итоге, параметром (размер тайла), практический выбор которого можно производить в соответствии с табл. 2.8-2.10.

РАЗДЕЛ 3
УСКОРЕНИЕ ВЫЧИСЛЕНИЙ МЕТОДА ДИСКРЕТНЫХ ТОКОВ (МДТ) ЗА СЧЁТ ВЕКТОРНЫХ РЕГИСТРОВ

В этом разделе исследуется возможность и эффект от совершенствования МДТ, ориентированного на применение векторных регистров ПК и возможность аналогичного усовершенствования других МДО. За последнее пятилетие практически доступный многим пользователям ПК размер векторных регистров увеличился до 256 бит, а количество связанных с ними операций системного уровня, а значит, и доступных программисту команд (FMA AVX), превышает несколько сотен. При таких условиях, теоретически, скорость вычислений с комплексными числами за счет использования этих регистров могла бы, увеличиться в 4 раза. Фактическое ускорение приложений должно быть меньше, но этот подход использует другой ресурс, чем тайлинг, не уменьшая его эффекта.

Альтернативные средства повышения степени параллельной обработки данных на ПК - карточки CUDA и платы ускорения - влекут, как выяснено в первом разделе, прямые дополнительные расходы и другие сложности. Векторизация вычислений, ориентированная на векторные регистры, достаточно нетривиальная потому невозможна способом непосредственной компиляции. Она требует изменений в порядке вычислений, а затем - применения нетривиальных решений в программной реализации. Излагаемый материал базируется на работах [64, 67].

3.1 Обоснованность использования векторных регистров в реализации МДО на ПК

Метод дискретных токов (имеющий варианты в зависимости от поляризации поля и выбранного фундаментального решения уравнения Гельмгольца) исходит из той или иной системы уравнений первого рода (для функций плотности тока на контурах) с сингулярными ядрами типа Коши,

ядрами типа логарифма или гиперсингулярными ядрами [23]. Такая система получает дискретную модель в соответствии с квадратурными формулами для соответствующих интегралов (понимаемых при необходимости в смысле обобщенных функций) [16], и их приближенное решение сводится к решению СЛАУ. При этом вычислительные трудности возникают для сложных систем в резонансном диапазоне волн, особенно при большом числе рассеивающих контуров. Примером может служить конфигурация на рис. 1 (поперечное сечение). Если электродинамическая

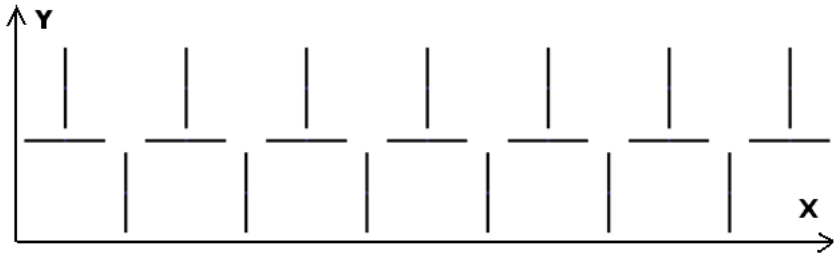


Рис. 3.1 – Пример решетки, состоящей из 20 металлических лент

система насчитывает десятки элементов, например, лент, а волновые числа берутся, скажем, от 10 до 20, то для обеспечения необходимой точности параметр дискретизации, а с ним - размерность матрицы должны принимать большие значения (размерность матрицы в таких задачах достигает нескольких тысяч). Матрицы таких систем обычно умещаются в памяти современных персональных компьютеров (ПК), но время решения может длиться часами.

При решении СЛАУ, полученного дискретизацией задачи (1)-(6) (по МДТ или другим методом, использующим граничные уравнения), необходим такой обмен данными между параллельно выполняемыми процессами, который требует их постоянного взаимодействия через общую память. Это обстоятельство обуславливает нетривиальность использования специальных особенностей архитектуры компьютерной системы. К тому же, классический подход к распараллеливанию в задачах линейной алгебры типа решения СЛАУ с заполненной матрицей состоит в поблочной работе с матрицей.

Поэтому, как объяснялось выше, потенциальные преимущества, связанные с большим числом процессорных ядер, которые имеются в CUDA, в данном случае полностью не реализуемы (в связи с постоянной загрузкой и выгрузкой данных из оперативной памяти в память карты CUDA).

Мы ставили перед собой задачу ускорить выполнение вычислений по МДТ за счёт векторных регистров. Для сравнимости результатов счёта, он проводился для нахождения стандартного минимального численного решения (СМЧР) дифракционной задачи. Это означает нахождение «дискретных токов» на основе дискретной модели системы граничных уравнений и построение поля в дальней зоне (диаграммы направленности с шагом 1˚для напряженности поля в соответствии с данной поляризацией).

Таблица 3.1.

Трансляция функции с языка C++ в ассемблерный код

C++ код	Код, сгенерированный компилятором

double VecMul(double* a, double* b, int size)
{
double sum = 0;
for (inti=0; i< size ; i++)
{
sum += a[i] * b[i];
}
return sum;
}
	VecMul(double*, double*, int):
	vxorpd %xmm0, %xmm0, %xmm0
	testl %edx, %edx
	jle .L4
	xorl %eax, %eax
	.L3:
	Vmovsd (%rdi,%rax,8), %xmm1
	vfmadd231sd (%rsi,%rax,8), %xmm1, %xmm0
	addq$1, %rax
	cmpl %eax, %edx
	jg .L3
	ret
	.L4:
	ret

На текущий момент компиляторы, казалось бы «умеют» использовать векторные регистры. Однако, как они это делают, показывает сравнение исходного кода на языке C++ с ассемблерным кодом, созданным компилятором GCC-4.9.

Разработав тест на перемножение двух векторов, видим (табл.3.1), что в векторные регистры попало только одно вещественное значение удвоенной точности, то есть из 256 бит задействовано только 64 бита.

Поясним, что в таблице справа vfmadd231sd() - это специальная ассемблерная инструкция, соответствующая оператору “sum += a[i] * b[i]; “.

Как видно из этого примера, использование векторных регистров компилятором сводится на сегодня к простому расширению количества доступных регистров, что дает при выполнении ускорение порядка нескольких процентов. Для большего ускорения за счёт использования таких инструкций можно применять ассемблерные вставки с кодом, написанным вручную. Однако предпочтительнее так называемые intrinsic-функции - специальные системно-зависимые функции для реализации общеупотребительных операций, которые намного эффективнее стандартных за счёт использования особенностей компьютерной архитектуры на уровне ассемблерного кода [85].

3.2 Векторизация МДТ для ПК

Наиболее затратные по времени этапы задачи моделирования взаимодействия электромагнитных волн с металлическими экранами по МДТ – это заполнение матрицы, решение СЛАУ и в некоторых случаях расчет необходимых характеристик. Исходя из этого на рис. 3.2 представлена схема векторизации МДО.

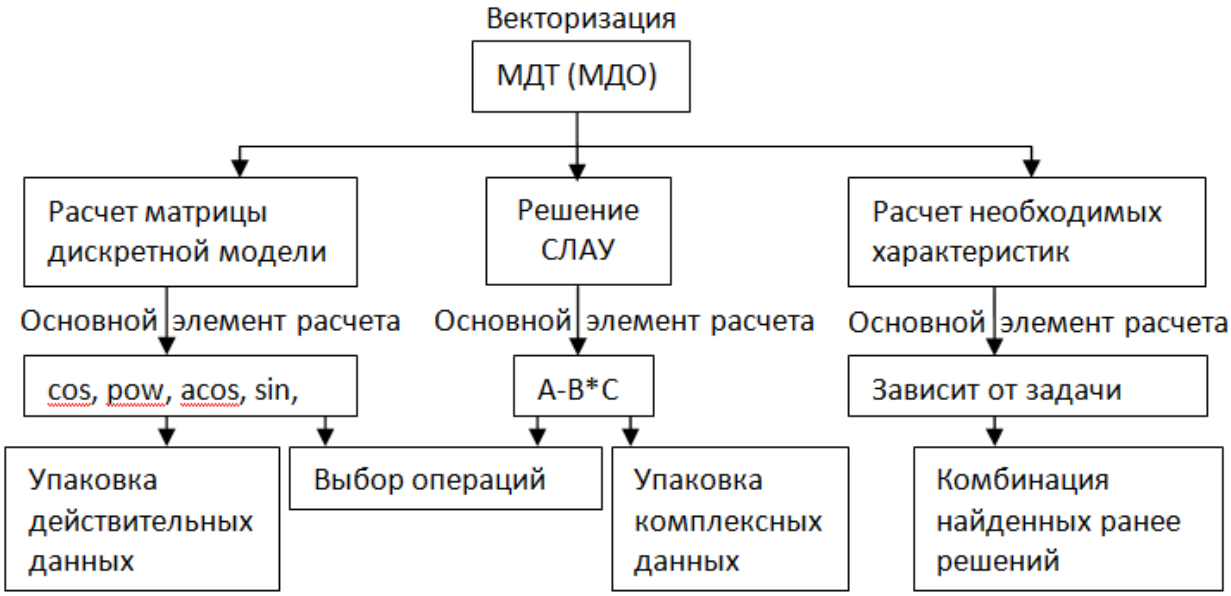


Рис. 3.2 – Схема векторизации МДО с выделением основных по времени расчета операций

Нетрудно вычислить, сколько действительных и сколько комплексных чисел следует размещать в векторный регистр (в диссертации подробно рассмотрены регистры по 256 бит, также брались во внимание другие возможности, и поэтому программные реализации требуют только перенастройки при изменении длины регистров). Однако существует далеко не один вариант размещения чисел, особенно комплексных, если речь идет о комплексных числах. Согласно выбранной модели исполнителя вычислений с регистрами лучший по количеству нетривиальных операций при упаковке и распаковке в векторные регистры вариант, показан на рис. 3.3.

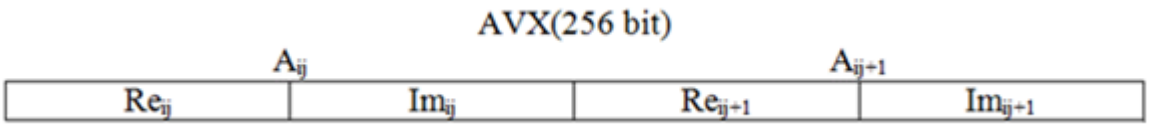


Рис. 3.3 – Упаковка комплексных чисел в векторный регистр

Векторизация МДТ, рассчитанная на максимальный эффект, требует ручного написания кода, в свою очередь, это влечем перебор последовательностей различных инструкций для реализации типовых

(многократно повторяющихся в циклах МДТ) сложных операций (часто с комплексными значениями), таких как A - B * C. При этом инструкции (intrinsic-функции AVX / FMA) многократно дублируются по основным исходным значением, имея различные побочное действия, которые могут использоваться. Ручной подбор оптимального по времени выполнения варианта был бы слишком длинным или нуждался бы в отдельном программировании с тестированием и настройкой, что тоже требует времени. Прием, который ранее упростил решение подобной задачи при построении тайлингового метода, применить невозможно, поскольку нельзя предсказать относительное время выполнения инструкций, тем более, их групп (для отсечения ветвей поиска). Нами была выдвинута гипотеза о том, что компилятор с языка C++, который использует достаточно хорошую теорию оптимизации, находит оптимальную или квазиоптимальную последовательность инструкций для реализации основных элементов расчета, учитывая расширение AVX/FMA, хотя результат компиляции и не реализует содержательные вычисления. Поэтому процедура практической оптимизации была такова:

1. сначала компиляция специального фрагмента C++ кода, имитирующего нужную операцию,
2. дизассемблирование скомпилированной программы
3. ручной перенос найденной компилятором последовательности инструкций в контекст нужного вычисления с содержательно заполненными векторными регистрами,
4. экспериментальная проверка квазиоптимального характера найденного решения по времени выполнения.

Эксперимент заключался в сравнении подсказанного компилятором решения по некоторым множествам других решений. Попытка сравнения с максимально полным множеством решений наталкивается не только на количественный фактор, но и на нечеткую определенность множества допустимых решений. Для выражений, выполнение которых надо было

оптимизировать, решения, подсказанные компилятором, оказались экспериментально лучше.

Из соображений сохранения других резервов ускорения счёта описанных в первом и втором разделах решение СЛАУ с комплексной матрицей опирается на компактную схему Гаусса. Мы, при использовании векторных регистров, сталкиваемся с определёнными трудностями:

- работа с участками памяти, выделенными под строки треугольной матрицы, не согласуется с естественным требованием, которое состоит в пропорциональности длины таких участков 32 байтам (для AVX),
- нет встроенных операций над комплексными числами, которые используют векторные регистры, и такие операции следует разрабатывать самостоятельно,
- необходимость применять так называемое выравнивание данных в памяти.

Блочное представление матрицы, которое было разработано для тайлингового метода, при кратности размера квадратных блоков (оптимального в смысле [49]) 32 байтам, позволяет справиться с первым и третьим из указанных осложнений, если не считать диагональных блоков.

Решение СЛАУ с помощью компактной схемы метода Гаусса [12] требуют циклического использования операций умножения и вычитания комплексных чисел, которые, находясь в векторных регистрах, должны быть, сведены к действиям с парами вещественных чисел. Рассмотрим подробно первой из этих операций, которая при практической реализации на ПК отнимает больше времени. Сложность в том, что в этом процессе необходимо переставлять местами вещественные компоненты числа внутри одного регистра. Несмотря на то, что эта проблема вычислителям понятна, её конструктивное решение для AVX регистров, судя по источникам, которые удалось обнаружить, на данный момент отсутствует. Поэтому имеет смысл изложить наше решение.

Компилятор языка C++ стандартным образом представляет комплексное число в памяти ПК в виде двух последовательно расположенных чисел типа double (8 байт). Те же 16 байт представления комплексного числа мы помещаем в векторный регистр. Вспоминая также об операции сложения/вычитания, целесообразно определить совмещённую операцию mulAndSub(), которая обеспечит действие с тремя комплексными числами, имеющее смысл оператора A = A - B * C.

Реализация этой операции опирается на описанную выше процедуру практической оптимизации. В результате нами разработан следующий код с использованием векторных регистров:

```
void mulAndSub(cmpx* to, cmpx* from1, cmpx* from2)
{
__m128d* v1 = (__m128d*)from1;
double* v2 = (double*)from2;
double* v3 = (double*)to;
    __m256d ymm0_v1;
    __m256d ymm1_v2;
    __m256d ymm2_axa;
    __m256d ymm3_perm_v2;
    __m256d ymm4_axb;
    __m256d ymm5_bb_xch_ab;
    __m256d ymm6_aa_xch_ab;
__m256d ymm7_res;
ymm0_v1 = _mm256_broadcast_pd(v1);
    ymm1_v2 = _mm256_loadu_pd(v2);
//mul
ymm2_axa = _mm256_mul_pd(ymm0_v1, ymm1_v2);
    ymm3_perm_v2 = _mm256_permute_pd (ymm1_v2, 0b0101);
    ymm4_axb = _mm256_mul_pd(ymm0_v1, ymm3_perm_v2);
```

```
        ymm5_bb_xch_ab = _mm256_unpackhi_pd(ymm2_axa, ymm4_axb);
        ymm6_aa_xch_ab = _mm256_unpacklo_pd(ymm2_axa, ymm4_axb);
        ymm7_res=_mm256_addsub_pd(ymm6_aa_xch_ab, ymm5_bb_xch_ab);
    //sub
    __m256d ymm9_v3 = _mm256_loadu_pd(v3);
        ymm9_v3 = _mm256_sub_pd(ymm9_v3, ymm7_res);
        _mm256_storeu_pd((double*)to, ymm9_v3);
    }
```

Таблица 3.2.

Ускорение вычислений СЛАУ при использовании тайлингового параллельного векторизованного метода

N	t1: базовая реализация	t2: тайлинг и многопоточность	t3:+векторные регистры	Ускорение счета t3/t1
1000	5.17	0.88	0.33	15.67
1731	28.71	3.97	1.25	22.97
2996	181.84	20.06	5.7	31.90
5186	1175.29	101.25	28.31	41.51
8978	6502.14	525.49	143.42	45.34

Матрица дискретной модели МДО, состоящая из комплексных чисел, должна быть сформирована так, чтобы решение СЛАУ могло быть выполнено с помощью векторных регистров, как было нами описано. Процессы вычисления матричных элементов включают вычисления и элементарных функций, и специальных функций математической физики. Это существенно отличает этап формирования матрицы от решения СЛАУ. В частности, при заполнении матрицы дискретной модели мы сталкиваемся с необходимостью отображать на векторные регистры известные методы

вычисления таких функций, как функции Ханкеля. Из этих соображений на данном этапе вычислений МДО может оказаться более выгодным представить процесс как процесс вычисления в векторных регистрах вещественных значений и лишь в конечной фазе придавать результатам комплексную форму, ориентированную на решение СЛАУ. При этом интерпретация расчётных формул исходного МДО для вычислителя с векторными регистрами ещё более неоднозначна, чем в случае со СЛАУ, поскольку набор AVX/FMA векторных инструкций процессора поддерживает не только все базовые арифметические и логические операции, но также элементарные математические, в частности, тригонометрические функции. Оценив, что большая часть времени вычисления элементов матрицы МДО действительно уйдёт на расчёт вещественной и мнимой частей комплексных числа по отдельности, нами использован для МДТ и рекомендуется для других МДО расщеплять работу комплекснми значениями на два процесса обработки вещественных, за исключением тех случаев (в заключительной фазе), когда требуется проведение над комплексными числами только арифметических операций. Поэтому при вычислении элементов матрицы дискретной модели наряду с упаковкой в векторные регистры комплексных чисел, использовалась, также следующая упаковка мнимых и вещественных частей комплексных значений, имея в виду их отдельную обработку (рис. 3.4).

AVX(256 bit)

Re_{ij}	Re_{ij+1}	Re_{ij+2}	Re_{ij+3}	Re_{ij+4}	Re_{ij+5}	Re_{ij+6}	Re_{ij+7}

AVX(256 bit)

Im_{ij}	Im_{ij+1}	Im_{ij+2}	Im_{ij+3}	Im_{ij+4}	Im_{ij+5}	Im_{ij+6}	Im_{ij+7}

Рис. 3.4 – Упаковка комплексных чисел в векторные регистры, используемая при заполнении матрицы дискретной модели.

3.3. Использование векторизации в других вычислительных МДО

Описанные выше решения по векторизации МДТ не выглядят специфичными для этого метода, и в предыдущем подразделе общие соображения часто приводились по отношению к произвольным МДО. Однако эти соображения превращаются в неоспоримые факты только после их материализации в конкретной вычислительной схеме, после программной реализации и подтверждения достигнутого эффекта в испытаниях.

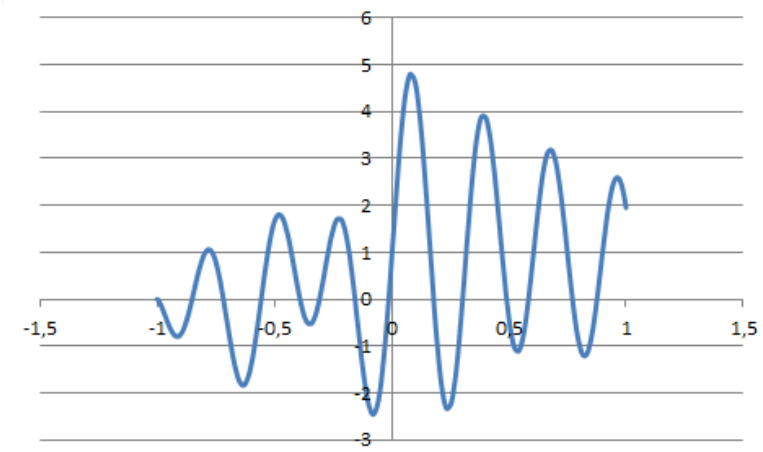
Первоначально векторизация позволила усовершенствовать МДТ в случае E-поляризации. Перенос отработанных при этом решений на другие МДО решения дифракционных задач оказался прозрачным благодаря двум обстоятельствам.

Первое то, что сам приём векторизации прост и прозрачен: если в цикле, состоящем из M повторений различные порции данных обрабатываются с помощью одних и тех же инструкций, то данные следует занести в векторный регистр так чтобы число упакованных порций имело суммарную длину в байтах равную длине регистра. Далее, обычно M зависит от параметров приближенного решения задачи, например, пропорционально параметру дискретизации N в МДО. Эти параметры подбираются так, чтобы M оказалось кратным числу обрабатываемых порций, которые умещаются в векторном регистре.

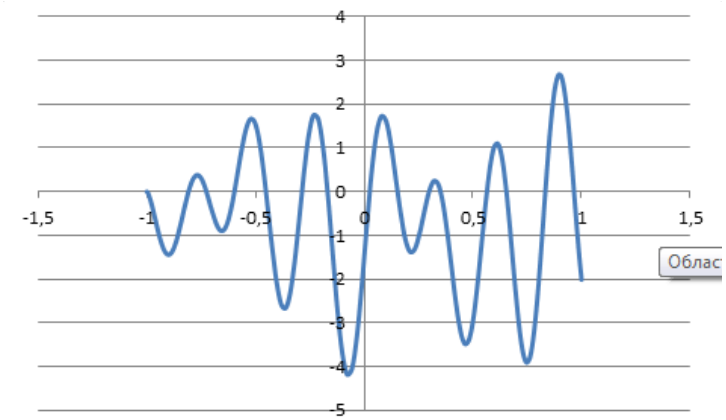
Второе, несмотря на то, что описанный общий приём бесполезен в большинстве конкретных задач, вследствие специфики выделения и упаковки порций данных в конкретных вычислительных методах, для МДО он срабатывает постольку, поскольку там сохраняет справедливость схема рис. 3.2. Тогда упаковка в регистры, описанная в подразделе 3.2 переносится в векторизуемый МДО без изменений, остаётся объединить в одно повторяемое действие отрезки циклов исходного метода, обеспечив, чтобы в этом действии обработка, соответствующая одному отрезку, происходила в векторном регистре.

По этому методу была легко выполнена (в объёме СМЧР) векторизация МДТ в H-случае. Большего внимания потребовала векторизация метода А. М. Назаренко для приближенного решения задач дифракции упругих волн на эллиптических включениях, описанного в подразделе 1.3, который имеет другую постановку СМЧР. Однако и в этом случае никаких новых решений при совершенствовании вычислительного метода не потребовалось.

3.4 Экспериментальная оценка ускорения вычислений за счет использования векторных регистров



А



Б

Рис. 3.5 – А – мнимая, Б – реальная часть значения дискретного тока на первом вертикальном контуре, длина волны 1/7, 20 контуров

В численных экспериментах, проведенных для оценки ускорения счёта, использовался процессор Intel core i5-4430. Его векторные регистры имеют размер 256 бит, что позволяет помещать в них 2 комплексных числа с плавающей точкой двойной точности. Сравнивались между собой разные реализации СМЧР по МДТ с ядром типа логарифма: базовая (один последовательный процесс), реализация на языке C++, оптимизированная по размеру зерна [45] и другим параметрам согласно [49] и модификация этой реализации под векторные регистры.

Для случая падения на электродинамическую структуру, изображенную на рис. 1, E-поляризованной волны вдоль оси Y под углом π к ней мы вычисляли интерполяционные приближения к плотностям токов на всех лентах (пример на рис. 3.5) и строили диаграмму направленности (пример на рис. 3.6) при значениях волнового числа к = 4π, 14π.

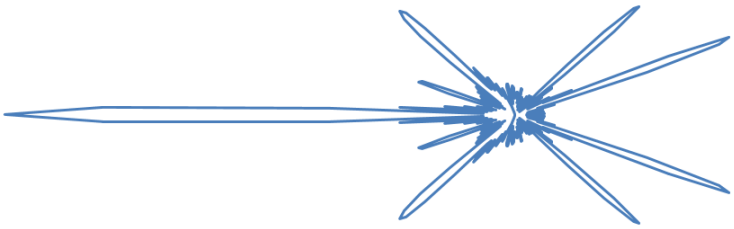


Рис. 3.6 – Диаграмма направленности (повернута на π/2), длина волны ½, 20 контуров

В случае к = 14π дискретная модель имела размерность 1000 (по пять дискретных особенностей на длину волны), а для проверки достигнутой точности вычислений параметр дискретизации дважды увеличивался примерно в $\sqrt{3}$. Аналогичные расчёты проводились для 80 контуров, причём дугообразных. Хронометраж вычислений МДТ с помощью указанных выше реализаций представлен на рис. 3.7. Отметим, что в трёх других примерах электродинамических структур, составленных из соосных цилиндрических

экранов, при совпадении размерности матриц продолжительность вычислений СМЧР по МДТ изменялась не более чем на 4%.

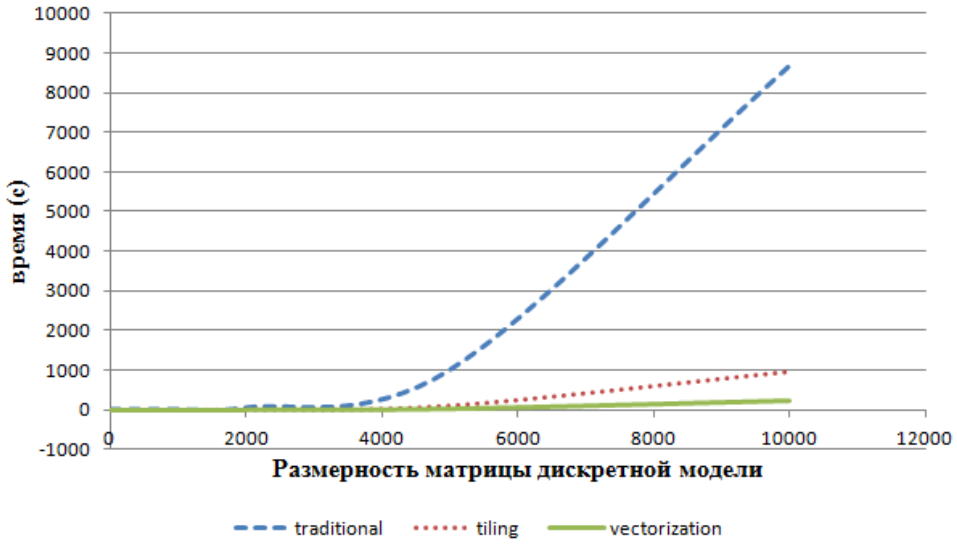


Рис. 3.7 – Зависимость времени выполнения (с) от размерности дискретной модели

3.5 Компьютерное моделирование взаимодействия электромагнитных волн с металлическими экранами

Для оценки полученного нами ускорения вычислений с использованием параллельного тайлингового метода и метода векторизации был проведен ряд экспериментов. Эксперименты проводились на примерах применения в содержательных задачах, которые ранее были исследованы другими авторами с использованием традиционных форм МДО:

- предканторовые цилиндрические антенны сложного профиля (К.В. Несвит, 2012),
- волноведущие системы эллиптических экранов (А. Носич , 2007).

При повышении порядка предканторовой структуры на пять порядков по сравнению с опубликованными примерами размерность дискретной модели получила значение N = 10250, а СМЧР было получено на 4-ядерном

ПК всего за 5 минут. Здесь и далее под 4-ядерным компьютером подразумевается Intel core-i5 4430 (3.0ГГц, 4 ядра, AVX/FMA), 8 Гб.

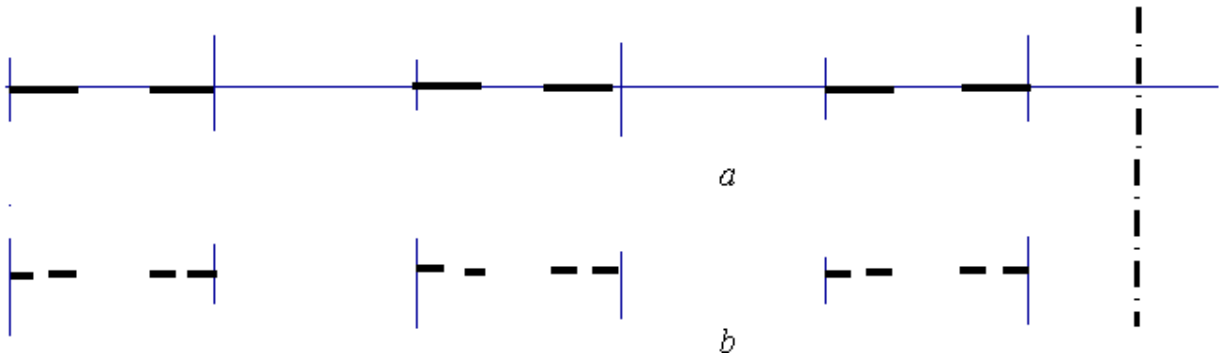


Рис. 3.8 – Левые половины симметричных предканторовых структур в сечении цилиндрических проводящих экранов, представляющие интерес с точки зрения построения широкополосных антенн: а) порядка 3; б) порядка 4.

Второй пример - задача о поле в средней зоне при взаимодействии излучения точечного источника E поляризованного поля длины волны λ с открытым волноводов (система эллиптических экранов, каждый из которых имеет длину примерно в 25λ). Длина волны сравнительно мала по отношению к длине сечений экранов, и идея этого устройства в том, что при очень малых длинах волн электромагнитная волна должна распространяться близко к тому, что вытекает из законов оптики. Тогда луч, выходящий из источника, помещённого в первый фокус первого из экранов, обязан пройти через второй фокус, который совмещён с первым фокусом второго экрана и т.д., пока во втором фокусе очередного экрана не окажется преемник волнового пучка (рис. 3.9). В отличие от сильно идеализированного приближения геометрической оптики, реальный пучок будет размыт. Однако, если большая часть переносимой им энергии сосредотачивается в объёме, который можно уместить в растре приёмника, волновод работает (с потерями, зависящими от степени размытости пучка). Как показала, в частности, работа [89], можно обойтись без многочисленных и недешевых

натурных экспериментов, имеющих цель оптимизировать параметры подобной волноведущей системы, заменив их многочисленными компьютерными экспериментами. Во что обходятся такие эксперименты (по времени счёта)?

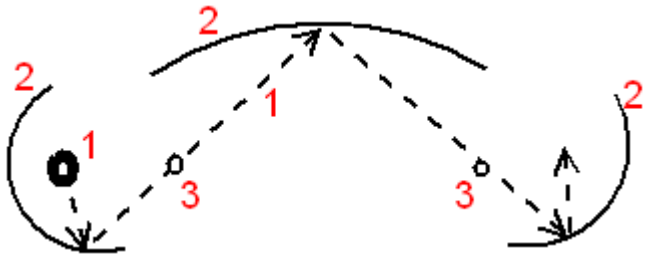


Рис. 3.9 – Пример открытого волновода, составленного из цилиндрических проводящих экранов эллиптической формы:
1 – точечный источник поля;
2 – проводящие экраны;
3 – фокусы эллипсов.

При размерности дискретной модели N = 1680 числовое решение по традиционному МДО отнимало 20 минут на ПК с параметрами 2.6 ГГц, 2 ядра, 1Гб. Вычисления на современном 4-ядерном ПК считаются в 5 раз быстрее и должны забрать 4 минуты. Тайлинговому параллельному векторизованному вычислительному методу на ту же работу понадобилось 2,5 сек., А за 20 сек. он успевает сделать вычисления с удвоенным значением размерности N = 3360.

Выводы по разделу 3

Усовершенствован МДТ в направлении его векторизации, которая позволяет применить AVX команды процессора, если они обеспечены в ПК, с целью использования векторных регистров для уменьшения времени расчета матрицы, решение СЛАУ дискретной модели предельного уравнений и вычисления диаграмм направленности, причем указано прозрачный метод переноса этой векторизации на другие МДО решения дифракционных задач.

Ускорение МДО вычислений способом векторизации позволяет ускорить их порядка 3 раз, но этот эффект не зависит от применения тайлинга, и в сочетании с ним ускорение в типичных примерах происходит в среднем примерно в 20 раз.

РАЗДЕЛ 4
ОРГАНИЗАЦИЯ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ МДО, СОГЛАСОВАННАЯ С ВАЖНЕЙШИМИ КРИТЕРИЯМИ ЕЁ ПРАКТИЧНОСТИ

Распространение параллельного тайлингового векторизованного метода решения МДО для задач, отличных от СМЧР, и на МДО, не рассмотренных в диссертации, потребует при некоторых обстоятельствах, доли творческой работы, а при организации компьютерного моделирования - принятия дополнительных решений по управлению потоками ОС. Обычно прикладному программисту лучше выполнять это средствами высокого уровня. Поэтому в первых экспериментах был использован высокоуровневый язык параллельного программирования Ада. Но лучшие по скорости выполнения решения удалось получить на более низком уровне, пользуясь средствами языков системного программирования C/C++. Поэтому возник вопрос о методе организации компьютерного моделирования дифракционных процессов, согласованном с объективными требованиями к программной реализации и удобством организации моделирования за счет высокоуровневых средств программирования. В связи с этим, в четвертом разделе исследуется соответствие указанным конкурирующим требованиям существующих информационных (программных) технологий. Излагаемый материал базируется на работе [51].

4.1 Важнейшие критерии практичности метода организации параллельных вычислений МДО и метод достижения компромисса

В предыдущих разделах работы, заботясь о создании методов, которые на современном ПК должны давать максимальное ускорение вычислений за счет тайлинга и наличия векторных регистров, мы оставляли на втором плане важные практические требования, которые возникают в связи с перспективами внедрения данных методов практику компьютерного моделирования дифракционных процессов. Прежде всего, речь идёт о

простоте разработки реализующих программ. Этому аспекту уделялось внимание постольку, поскольку речь шла о переносе решений, полностью разработанных для МДТ, на другие задачи дифракции, и можно было указать на относительную простоту или сложность использования образцов и методов их применения, на наличие или отсутствие необходимости проявлять при этом творчество. Однако наибольшей ответственности при этом требует обеспечение синхронизации вычислительных процессов (соответственно, - потоков ОС), если оно необходимо (что связанно обычно с оригинальностью постановка дифракционной задачи). Ясно, что упрощение программной реализации в этом аспекте влечёт менее удачную синхронизацию и, тем самым, - затягивает время выполнения. Наоборот, стремление добиться максимального ускорения при выполнении заставляет искать всё более сложные решения программной реализации, как это видно было в подразделах 2.1, 2.3, 2.4, 3.2. Итак, скорость и простота реализации, вообще говоря, - конкурирующие требования.

Существует ещё одно, не столь очевидное, но не менее важное с точки зрения решения сложных задач дифракции требование – это обеспечение динамического распределения ресурсов. Например, при автоматическом контроле точности расчётов путём сравнения их результатов, полученных при разных значениях параметра дискретизации, целесообразность закрытия одной системы потоков и создания новой вполне очевидна. На самом деле даже при нахождении СМЧР рассмотренных задач, выполнение компьютерного моделирования можно ещё несколько ускорить за указанный счет, но самостоятельно программировать динамическое распределение ресурсов прикладной программой сложно и чревато скрытыми ошибками. Противоречие между требованиями просты реализации и динамичности распределения настолько остро, что согласие последнего требования с требованием ускорения выполнения, как правило, не является решающим.

Ситуация напоминает задачи многокритериальной оптимизации, но практически неформализуема. Соответственно, вместо формальных методов

мы используем метод естественного компромисса. А именно, если и синхронизацию и динамическое управление ресурсами можно переложить на существующую ИТ общего назначения, то это удовлетворит 3-й критерий и наверняка заметно улучшит второй (упростит программирование). Останется проверить, не слишком ли при этом замедляется выполнение моделирующих дифракцию приложений (расчёт тот, что замедление, неминуемое при использовании инструментария общего назначения, частично компенсируется за счет динамичности распределения ресурсов).

4.2 Компромиссная тайлинговая векторизованная реализация МДТ

Выдвинем гипотезу о том, что реализация тайлингового параллельного МДО средствами выбранной технологии уступает реализации средствами языков C/C++ (по ускорению выполнения) не более чем на 10%, и проверим её на задачах нахождения СМЧР по МДТ. Выбор рубежа в отсутствие прецедентов субъективен, но выглядит разумным.

Были рассмотрены альтернативные варианты выбора технологии программирования, которые соотносились с OpenMP [95, 81]:

- MPI - замедленное выполнение (по сравнению с OpenMP) при наличии взаимодействующих процессов [35, 61, 72];
- Intel threading Building Blocks [98] - требует значительной модификации кода;
- Intel Cilk Plus [77] - очень похож на OpenMP, но поддерживается меньшим количеством компиляторов.

На основе этих сравнений выбрали OpenMP. Применённую нами технику распараллеливание с помощью OpenMP лучше пояснить на предельно упрощенном примере. Пусть это будет умножение матриц.

При организации данных в программе используются двумерные динамические массивы:

```
double** A;
```

```
double** B;
double** C;
```

Исходный алгоритм, который мы представим на языке программирования C/C++ имеет вид:

```
for (int i = 0; i< size; i++) {
for (int j = 0; j < size; j++) {
for (int k = 0; k < size; k++) {
C[i][j] += A[i][k] * B[k][j];
      }
    }
      }                                                    (4.1)
```

Для превращения в OpenMP программу достаточно добавить одну строку:

```
#pragma omp parallel for shared(A, B, C)
```

Эта строка сообщает компилятору, что следующий далее цикл можно распараллелить, считая массивы в памяти A, B и C общими ресурсами. Под общим ресурсом здесь подразумевается, что каждая операция внутри цикла, связанная с этими массивами переменных должна выполняться над ними, а не над их локальными копиями в процессах. При отсутствии декларации shared(A, B, C) в каждом потоке создалась бы копия этих переменных. Хотя это и не критично в данном примере, но в приложении МДТ при использовании данных со стека привело бы к получению неправильных результатов. В итоге реализация матричного умножения с параллельной обработкой получает вид:

```
#pragma omp parallel for shared(A, B, C)
```

```
for (int i = 0; i< size; i++) {
for (int j = 0; j < size; j++) {
for (int k = 0; k < size; k++) {
C[i][j] += A[i][k] * B[k][j];
        }
      }
}
```
(4.2)

Посмотрим, каким образом это сказывается на ускорении вычислений (табл. 4.1).

Таблица 4.1.

Результаты тестирования параллельной на базе OpenMP (4.2) и исходной (4.1) реализации демонстрационного примера

Размерность матриц	1000	2000	4000	10000
Время выполнения фрагмента (4.1)	2,3с	21,46с	274с	6947
Время выполнения фрагмента (4.2)	8,1с	84с	1091с	1744
Коэффициент ускорения при 4-х ядрах	3,52	3,91	3,98	3.98

Ускорение выполнения вычислений, в которых можно выделить 4 слабо взаимодействующих процесса, теоретически может на 4-ядерном процессоре достигать значения 4,0. Использование OpenMP позволяет с ростом размерностей перемножаемых матриц достигать максимального теоретического ускорения с незначительным недобором (0.5%). Однако мы видим, что при средних (с точки зрения современных компьютерных вычислений) размерностях матриц «накладные расходы» по времени могут составить около 14%, что заставляет ставить задачу об обязательном исследовании этого вопроса для параллельных реализаций более сложных вычислений, когда взаимодействие процессов носит систематический характер.

4.3 Библиотека SDCM ускоренных вычислений МДО

При построении библиотеки компонент SDCM (Speed Discrete Currents Modeling) решались две задачи.

Первая - разработка проекта библиотеки компьютерного моделирования дифракции электромагнитных волн на идеально проводящих экранах (в 2D постановкой задач), которая позволяет без специальных знаний или долгого своего изучения организовывать и проводить вычисления, существенно ускоряемые за счёт многопоточности процесса, мультиядерности процессора, оптимизации работы с кэшем памяти и векторных регистров процессора.

Второе – верификация и оценка характеристик тех приложений для вычислительных экспериментов по моделированию дифракционных процессов, которые можно собрать из модулей данной библиотеки.

Важнейшие модули библиотеки, названной при её оформлении SDCM, были фактически уже разработаны на языке программирования C++ в процессе разработки модификаций МДТ для использования эффектов тайлинга и параллельных вычислений. Однако они для своего понимания и использования в качестве компонент требуют от пользователя, который организует вычисления, свободного владения низкоуровневыми средствами управления параллельными процессами. Такой квалификации у специалистов по моделированию дифракции ожидать нельзя.

Поэтому в данной работе архитектура и основные системные решения разработаны на базе открытого стандарта для распараллеливания программ OpenMP. Это - высокоуровневое средство распараллеливания, которое позволяет автоматизировать управление потоками приложения. В то же время, при переходе на него не меняется однопоточный код. Реализация этого стандарта хорошо зарекомендовала себя у прикладников понятностью команд и простотой использования при удовлетворительной эффективности разработанных приложений [95].

Верификация модулей библиотеки SDCM основана на сравнении получаемых результатов моделирования с результатами работы традиционных реализаций МДТ. Наблюдается совпадение до 7-го знака мантиссы при вычислении значений дискретных токов.

Оценка эффективности по скорости работы проводилась на основе серий вычислительных экспериментов с разными реализациями в разных режимах. При этом оценивалась предполагаемая потеря эффективности как плата за удобство настройки готовых компонент при создании приложений компьютерного моделирования.

На диаграмме компонентов рис. 4.1 показана система модулей библиотеки SDCM.

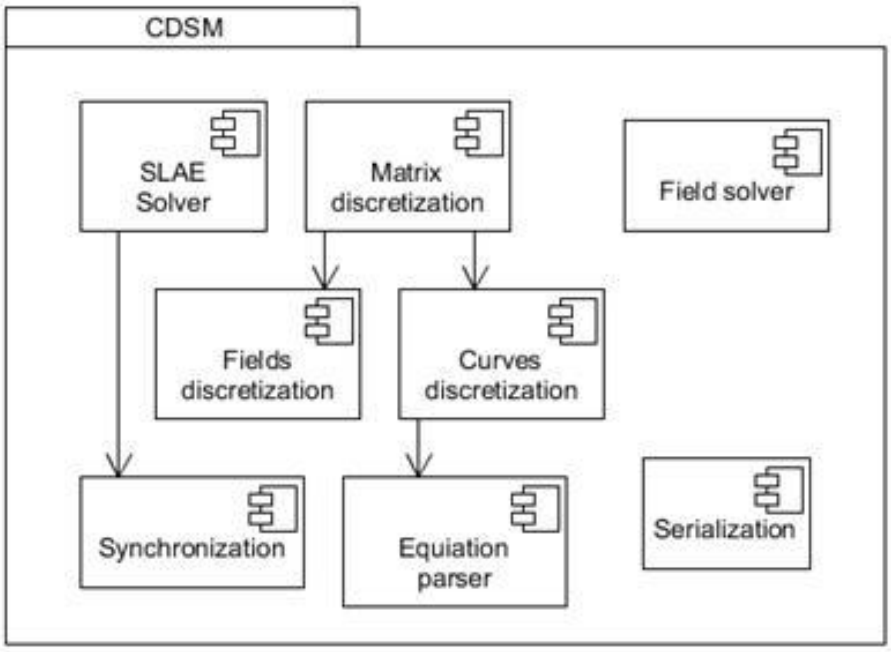


Рис. 4.1 – Модульная архитектура библиотеки SDCM

Сведения о назначении и размерах модулей собраны в табл. 4.2.

Таблица 4.2

Общие сведения о реализации компонентов в библиотеке SDCM

Модуль:	Краткое описание:	LOC:
SLAE Solver:	решение СЛАУ выч. методом компактной схемы Гаусса в вариантах:	155

	- стандартный, - модифицированный для оптимизации работы с кэш памятью, - параллельная модификация оптимизации работы с кэш памятью, - параллельная модификация с использованием векторных регистров при оптимизации работы с кэш памятью	
CurvesDiscretization:	сохранение информации о контурах в виде системы узлов вычислительного метода (МДТ)	140
FieldsDiscretization:	вычисление правых частей СЛАУ	70
MatrixDiscretization:	формирование матрицы МДТ: - естественный алгоритм, - модифицированный для оптимизации работы с кэш памятью, - параллельная модификация оптимизации работы с кэш памятью, - параллельная модификация с использованием векторных регистров при оптимизации работы с кэш памятью	120
FieldSolver:	вычисление диаграммы направленности и рассеянного поля в заданных точках	50
Synchronization:	при использовании OpenMP не используется	103
EquationParser:	аналитическое задание контуров	260
Serialization:	ввод-вывод по формату JSON	350

Распараллеливание заполнения матрицы и вычисления диаграммы направленности реализуется слабо взаимодействующими процессами и обеспечивается командой компилятору по обсуждённой выше схеме:

```
#pragma omp parallel for shared(matrix)
for(int i = 0; i<matrix_size; i++)
{
   for(int j = 0; j <matrix_size; j++)
   {
   matrix[i][j] = calculate_element(…);
   }
}
```

Распараллеливание блочного варианта компактной схемы метода Гаусса, которая используется для решения СЛАУ МДТ, с целью облегчения параллельных модификаций, осложняется систематическим и нетривиальным взаимодействием вычислительных потоков. Однако нами создана следующая реализация, состоящая из 3 частей:

1. вычисление диагонального i-ого блока;
2. вычисление i-ой строки из блоков подматрицы U;
3. вычисление i-ого столбца из блоков подматрицы L.

Вычисление каждого блока состоит из следующих этапов:

1. выбор блоков, от которых зависит текущий;
2. определение границ блоков;
3. вычисление внутри текущего блока.

Вычисление внутри текущего блока состоит из этапов:

1. вычисление вклада от блоков, от которых зависят значения текущего;
2. вычисление значений текущего блока (для блоков подматрицы L состоит из 2 этапов, для повышения локальности памяти).

Вычисления блоков внутри i-ой строки и i-ого столбца независимы, соответственно мы можем достаточно просто их распараллелить. Причем вычисление i-ой строки и i-ого столбца также могут выполняться параллельно.

Этот подход реализует идею, состоящую в использовании диагональных блоков как элемента синхронизации. За счет того что их количество (M) намного меньше, чем количество остальных блоков (M^2), мы можем пренебречь их выполнением в однопоточном режиме. Отметим, что объём необходимых команд системы OpenMP снова невелик, а схема их внедрения в код логична:

```
for (int block = 0; block <block_number; i++)
{
   // middle block calculation

   //U blocks
   #pragma omp parallel for shared(matrix, i)
   //top blocks calculation

   //L blocks
   #pragma omp parallel for shared(matrix, i)
   //left blocks calculation
}
```

К сожалению, при этом не удаётся избегнуть того недостатка, что создание потоков привязано к окончанию вычисления каждого из диагональных блоков матрицы СЛАУ. Это влечёт дополнительные накладные расходы на поддержку параллельных процессов (в сравнении с базовым случаем слабо взаимодействующих процессов). Однако нами была замечена и использована следующая эмпирическая закономерность.

Гипотеза. При решении СЛАУ относительный *вклад* в общую продолжительность решения СЛАУ *накладных вычислительных расходов* на распараллеливание по указанной схеме *с ростом числа блоков* (M^2) *уменьшается*.

Действительно, эта общая продолжительность должна была бы быть примерно пропорциональной M^2, тогда как продолжительность накладных расходов, связанных с созданием потоков, предположительно пропорциональна M.

4.4 Оценки производительности компромиссных решений МДО

Организация тестирования эффективности вычислений по времени

Таблица 4.3.

Допустимое снижение скорости выполнения вычислений при использовании OpenMP

Размерность матрицы N экранов	Время t1: многопоточный тайлинговый и векторизованный метод	Время t2: + OpenMP	Δt – разница во времени	Δt/t1*100 – показатель снижения скорости (%)
2x100	0.022	0.022	0	0.00
2x200	0.156	0.16	0.004	2.56
5x100	0.108	0.114	0.006	5.56
2x500	1.892	1.935	0.043	2.27
5x200	0.625	0.652	0.027	4.32
10x100	0.405	0.429	0.024	5.93
2x1000	13.47	13.75	0.28	2.08
10x200	2.39	2.531	0.141	5.90
5x500	7.15	7.441	0.291	4.07
5x1000	53.313	55.494	2.181	4.09
10x500	31.95	33.86	1.91	5.98
10x1000	246.81	261.723	14.918	6.04

Тестирование проводилось на платформе

процессор: Intel core i5-4430
ОЗУ: 8GB
ОС: Windows 10
Компилятор: Mingw 4.9.4

Отметим, что строгие методы математического решения дифракционных задач не применяются при высоких волновых числах, когда на проводящих экранах умещаются многие десятки длин волн. Если на каждом из примерно равных 10 экранах умещается, скажем, 50 длин волн, а на длине волны размещается 7-8 дискретных особенностей, как это принято делать при практическом компьютерном моделировании, то мы имеем $N \approx 10 \times 400$. Т.о. последние два случая, показанные в таблице, выглядят маловероятными на практике, и увеличение времени выполнения приложений, собранных на базе библиотеки SDCM, разработанной с помощью OpenMP, навряд ли будет составлять более 5%.

4.5 Компромиссность метода для задач взаимодействия электромагнитных волн с металлическими экранами

Принципиальным результатом проведенного исследования является возможности сохранить эффективность компьютерного моделирования МДО, которую обеспечивают тайлинг и векторные регистры процессора, отказавшись от сложного (и тем ненадёжного с точки зрения не имеющих специальной квалификации пользователей) написания блока синхронизации, применяя высокоуровневые средства OpenMP.

Как видно из наших рассмотрений, использование OpenMP удовлетворяет требованиям динамичности и простоты реализации, поставленным в подразделе 4.1, но по величине ускорения не может полностью сравниться с ручным распараллеливанием в силу невозможности избежать при автоматизации динамической синхронизации, осуществляемой OpenMP, постоянного пересоздания потоков при решение СЛАУ. Однако

проведенные эксперименты с реализациями МДТ показывают, что это снижение величины ускорения (в сравнении с ручным написанием блока синхронизации параллельных процессов) не выходит за рамки намечавшегося рубежа в 10%, и на практике, предположительно, будет составлять не более 5%.

Эти выводы не учитывают дополнительного резерва автоматизации работы с векторными регистрами. Та же самая технология OpenMP позволяет использование векторных регистров для распараллеливания вычислений. Однако накладываются следующие ограничения: в вычислениях разрешено использовать только арифметические операции, так же разрешается использование функций, но они должны быть заранее описаны с помощью OpenMP и, опять таки, обязаны использовать только арифметические операции. Это не позволяет в большинстве случаев получить столь же ощутимый прирост скорости выполнения, как использование intrinsic-функций. Так при использовании компилятора gcc 4.9 с поддержкой OpenMP версии 4.0 на простом демонстрационном примере перемножения векторов, описанном в третьем разделе, коэффициент ускорения колебался в пределах 1,5-2 раз, что явно хуже использования intrinsic-функций.

Выводы по разделу 4

Усовершенствован метод организации параллельного компьютерного моделирования дифракционных процессов, основанный на граничных интегральных уравнениях и МДО, в направлении достижения средствами высокоуровневого программирования согласованности критериев динамичности распределения ресурсов компьютера, сохранения скорости вычислений и простоты программной реализации, позволяет, в частности, в достаточной мере реализовать потенциал быстрого выполнения разработанных в диссертации вычислительных методов, не требуя от прикладника-реализатора решения сложных задач системного программирования. Эксперименты показывают, что при таком методе

организации компьютерного моделирования дифракции снижение скорости выполнения (сравнивая с применением C ++ технологий) уменьшается всего на несколько процентов, тогда как, по метрическим оценкам, работа программирования сокращается примерно в 2.7 раза.

Прикладным результатом является построение базовой версии программной библиотеки компьютерного моделирования SDCM, которая, в отличие от близкой по назначению библиотеки ЭДЕМ [26, 71], делает применение МДТ в ОКР возможным не за счёт эвристического упрощения сложных задач дифракции, а за счёт форсирования внутренних резервов вычислительной мощности современных ПК.

РАЗДЕЛ 5
ОЦЕНКА КАЧЕСТВА РЕАЛИЗАЦИЙ ТАЙЛИНГОВЫХ ВЕКТОРИЗОВАННЫХ МДО НА ПК

Первая часть раздела (подразделы 5.1-5.3) посвящены обоснованию тезиса о том, что разработанные в разделе 2, 3 вычислительные методы могут быть реализованы «без особенных усилий программирования». Из метрических оценок этой части, можно так же сделать заключение о том, что дополнительные усилия программирования в разработанном методе организации компьютерного моделирования дифракции, который использует технологию OpenMP, в несколько раз упрощает усилия прикладного программиста. Во второй части (подразделы 5.4-5.6) содержится обоснование того, что программные реализации разработанных в диссертации вычислительных методов, требуют (при решении тех же задач дифракции) меньших затрат электроэнергии при своем выполнении, чем реализации традиционных (не параллельных, не тайлинговых) методов. Метрические оценки указывают на то, что эти затраты уменьшаются примерно в 2-10 раз. Материалы раздела основаны на работах [53, 68].

5.1 Работа программирования и лексические характеристики реализаций МДТ

Благодаря универсальности схемы МДО программа EPolarizationMT, в которой нашли реализацию указные выше методы ускорения вычислений и которая предназначена для решения задач дифракции E-поляризованных волн на цилиндрических идеально гладких проводящих экранах в 2D постановке, может быть использована в качестве образца для построения ускоренных реализаций других МДО. Мы ответим на вопросы: каковы затраты на модифицированную, «быструю» реализацию в сравнении с обычной реализаций МДО? трудно ли работать с такой модифицированной реализацией? можно ли сосредоточить в отдельных модулях все

существенные изменения, связанные с ускорением, и насколько это рискованно? Свои прогнозы мы обоснуем аналогией с приложением EPolarizationMT, оценив его соответствующие характеристики в сравнении с однопоточной версией EPolarizationST. В качестве метрик (методов оценки) будут применены метрики энергетического анализа [48, 50]. Для их оценивания необходимо представить EPolarizationMT, написанную на языке C++, в форме схемы ПС (СПС), к которой предъявляются известные требования. Данный язык не обеспечивает однозначного представления, оно обеспечивается UML моделированием системы. Для модулей СПС нужно отыскать N - длину в программных символах (ПСим), η - алфавит (точнее, мощность алфавита) использованных ПСим, а также разделение ПСим на операторы, помечаемые or, и операнды – od ($N = N_{or} + N_{od}, \eta = \eta_{or} + \eta_{od}$). Это даёт холстедовские объёмы модулей $V = N \log_2 \eta$. Метрика оценки трудности

$$D\wedge = N_2 \eta_1 / 2\eta_2 \qquad (5.1)$$

(по Холстеду) традиционно использовалась в программной индустрии как показатель понятности исходного текста модуля. Модули можно классифицировать по абсолютным значениям этой метрики [48]. Ещё необходима оценка числа формальных параметров η^*, после чего потенциальные объёмы корректно определяются как $V^* = (\eta^* + 2)\log_2(\eta^* + 2)$. Для групп блоков и модулей эта величина получается суммированием по всем блокам. Добавляя к холстедовскому объёму модуля объёмы интерфейсных модулей, от которых он прямо зависит, и, делая поправки на другие отношения с учётом порядка разработки, получим метрику объёма разработки $W \geq V$. Она позволяет прогнозировать ошибки кодирования и оценивать трудность модулей:

$$B = W / V_{cr} \quad (V_{cr} = 3Hd, Hd = 10^3 bit \cdot sym), \; D = W / V^*, \qquad (5.2)$$

реализуя идеи Холстеда и обобщая их для программ сложной архитектуры. По этой метрике модули имеет смысл ранжировать. Произведение $A = D \cdot V$ даёт метрику работы программирования для модуля, а сумма по модулям

распространяет её на всю систему. По данным об архитектуре системы определяется её спецификационная энергия [48] (во многих случаях для групп блоков, имеющих суммарный потенциальный объём V^*, эта энергия $E=(V^*)^3/\lambda^2$, где λ – уровень, присвоенный языку программирования). Сравнение для ПС работы и энергии предоставляет метрика

$$q=(E-A)/\max(E,A), \qquad (5.3)$$

- «относительное интеллектуальное тепло». Критерий энергетической сбалансированности состоит в том, что в десятичном представлении q первая значащая цифра $\neq 9$ [46]. Устойчивая (от версии к версии) энергетическая сбалансированность свидетельствует о зрелости разрабатываемой системы.

5.2 Архитектура исследуемой системы и потенциальные объёмы

Общая логика компьютерного моделирования с помощью МДО стационарных и волновых процессов с линейной математической моделью вытекает из унифицированной схемы решения таких задач. Необходимы математическая модель на основе граничных уравнений (обычно интегральных, но понимаемых в смысле обобщенных функций в виду особенностей ядра), дискретная модель граничных уравнений, вычислительный метод для расчёта элементов матрицы дискретной модели, метод решения СЛАУ, метод расчёта требуемых прикладникам характеристик или полей. Соответственно этому выглядит на рис. 5.1 эскиз архитектуры ПС EPolarization, которая реализует МДТ для задач дифракции плоских E-поляризованных волн на цилиндрических идеально проводящих экранах с любыми сечениями - гладкими несамопересекающимися кривыми. Эта ПС является базой двух приложений: EPolarizationST – однопоточная реализация (контрольная) и EPolarizationMT. Компонент Gaus имеет только зависимость S от SimpleLu, а в многопоточной зависимость M от MTBlockLU.

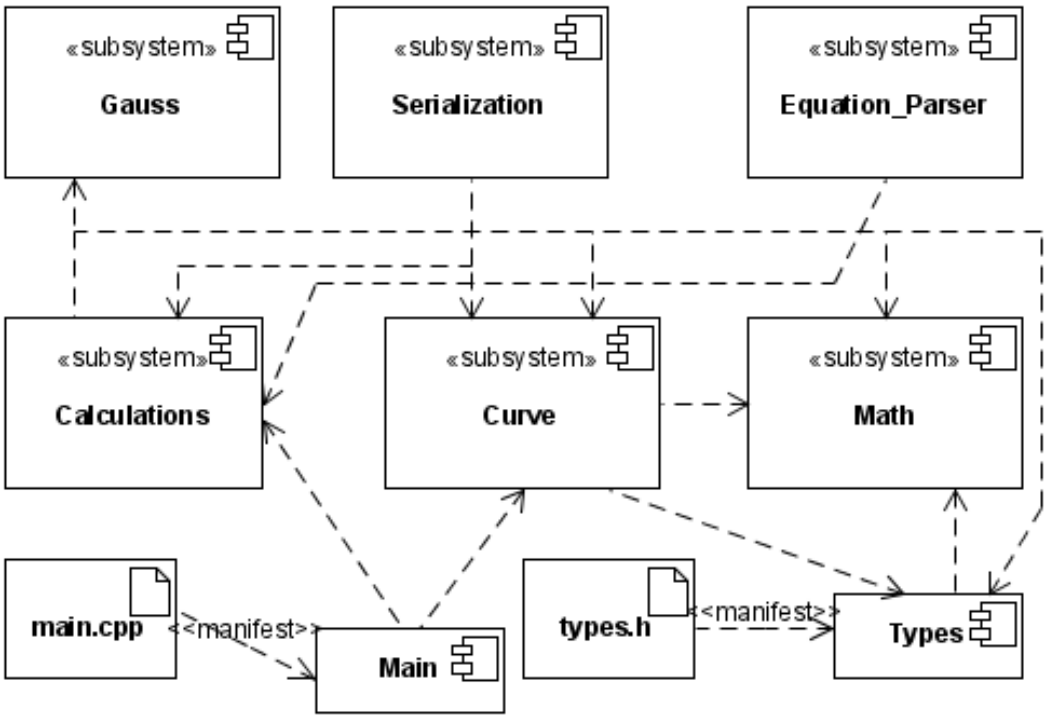


Рис. 5.1 – Эскизная схема ПС EPolarization (диаграмма компонент UML)

Раскрывая с помощью других диаграмм компонент архитектуру всех подсистем (пример на рис. 5.2) мы подойдем к получению схемы

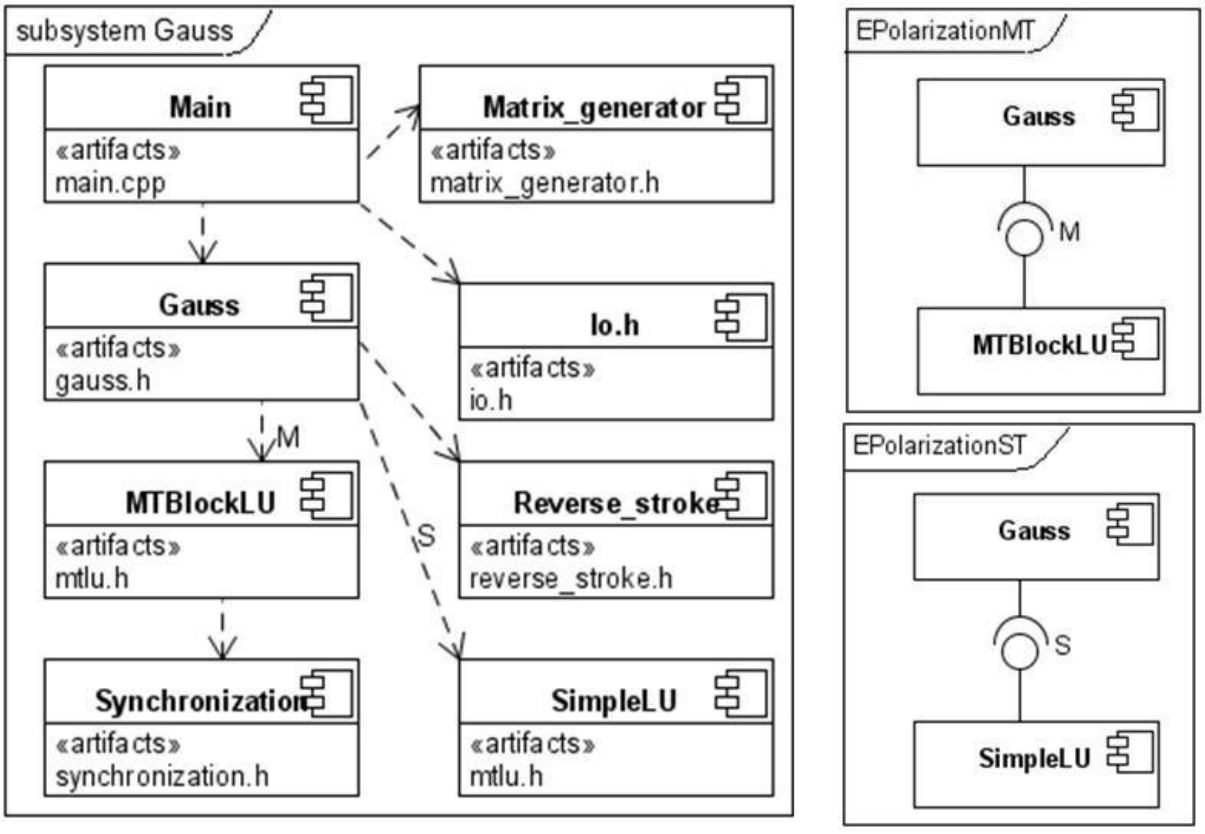


Рис. 5.2 – Компонент Gauss в однопоточной и многопоточной версиях

программной системы в смысле энергетического анализа. Для оценки СПС используется также следующая информация. Во-первых, характеристики интерфейсов, в частности, тех, на которые указывают отношения зависимости, показанные на диаграммах рис. 1-2 (и ещё на 5-ти, которые здесь не приводим). Во-вторых, внутренняя архитектура всех компонент, включая соответствующие операций ввода-вывода в телах интерфейсных модулей (в тех случаях, когда такие операции имеются). В-третьих, отношение «предшествия создания» между компонентами (которое в данном случае совпало с отношением стандартного умолчание для СПС).

На уровне исходных кодов все существенные отличия вариантов приложения сосредоточены в подсистеме Gauss (рис. 5.2), которая имеет только зависимость от интерфейса S, поддерживаемого вспомогательным компонентом SimpleLu, и зависимость M от интерфейса, поддерживаемого компонентом MTBlockLU. Технически это означает при переходе от обычной реализации к многопоточной замену компонента SimpleLU (рис. 5.3) на компонент MTBlockLU (рис. 5.4), что влечёт появление дополнительно ключевого модуля Synchronization и его тела (рис. 5.5).

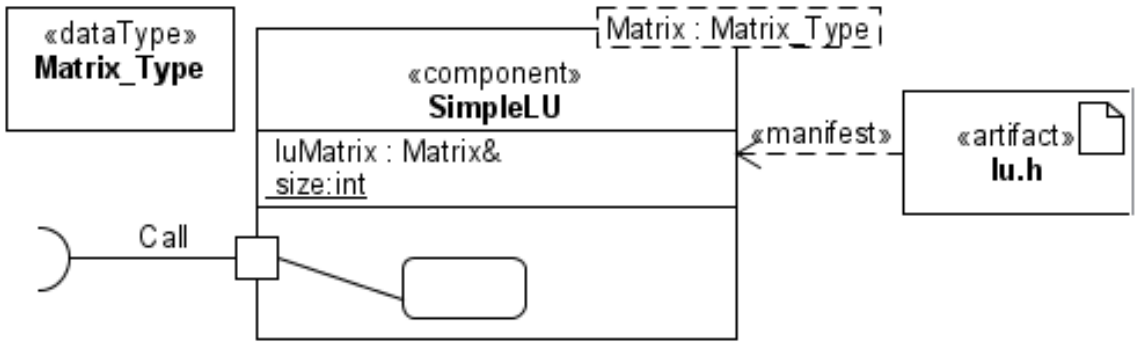


Рис. 5.3 – Альтернативная компонента подсистемы Gauss однопоточного приложения

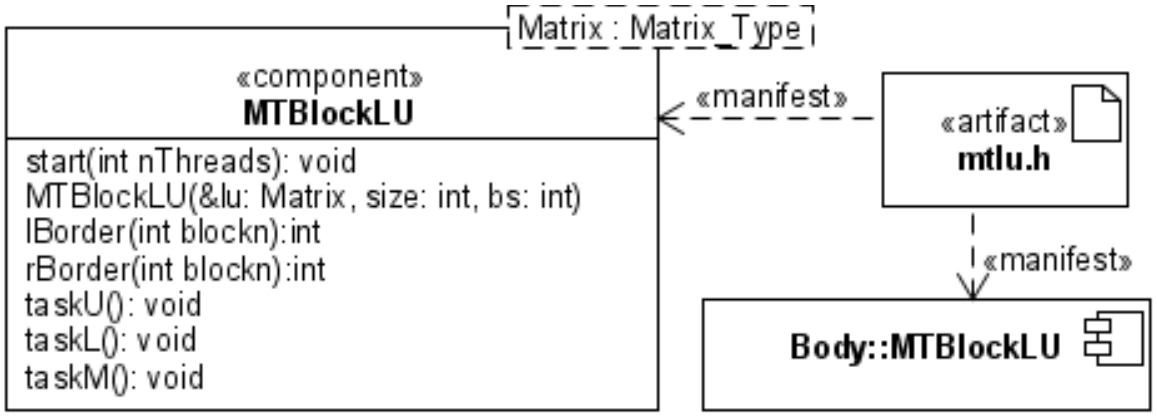


Рис. 5.4 – Альтернативная компонента подсистемы Gauss для приложения EPolarizationMT

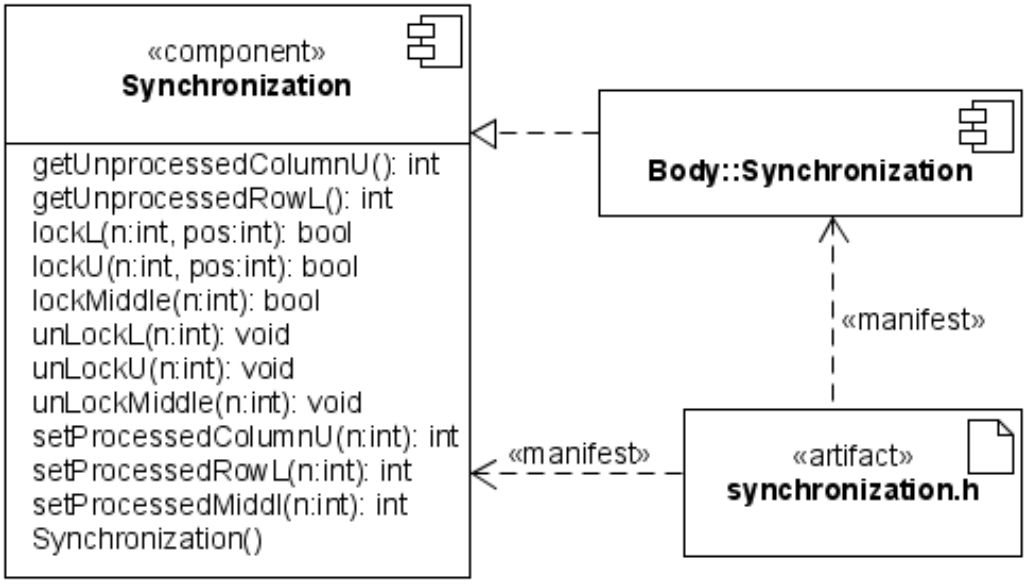


Рис. 5.5 – Дополнительные компоненты подсистемы Gauss в многопоточном приложении EPolarizationMT

5.3 Результаты энергетического анализа программной библиотеки

Распределение модулей по классам трудности D^ (1) дано в табл.5.1. Перечень наиболее трудных модулей по сумме номеров классов трудности D^ и D (2) - в табл.5.2, результаты энергетического анализа по системе в целом – в табл.5.3.

Таблица.5.1

Количество модулей в классах трудности D^ приложения EPolarizationMT

Классы по [46]:	0 класс	1 класс	2 класс	3 класс	4 класс	5 класс
модулей в классе	31	4	4	2	2	2

Таблица.5.2.

Наиболее трудные модули приложения EPolarizationMT

Модуль	Подсистема	Кл. D^	D^	Класс D	D
json_ui.cpp	Serialization	5	987	4	293
calc_manager.cpp	Calculations	5	435	4	318
grammar.h	Equation_Parser	3	263	5	668

Таблица.5.3.

Характеристики EPolarizationMT, полученные при энергетическом анализе

$\sum B_i$	$\max D_0^*$	$\min D_5^*$	$\max D_5^*$	A (kHd)	E (kHd)	q (3)
56	70,8	368	814	12,52	0,983	-0,92

Пусть B_i - оценка потенциального числа ошибок в i-м модуле на момент перехода от написания кода к систематическому тестированию, оценённая по объёму разработки (2). Эта оценка служит ориентиром, реалистичным обычно по первой значащей цифре [48]. Суммирование по всем модулям, в принципе, законно в силу использования при оценках именно объёмов разработки, а не Холстеда. Оно даёт ориентир 56 ошибок (табл. 3), а, значит, примерно столько же первичных дефектов этой разработки. Их успешное (без внесения существенного числа вторичных дефектов) исправление во многом зависело от трудности модулей системы. Используя для выяснения вопроса о трудности метрику её оценки

М. Холстеда по методу [46], мы на основании табл. 1 можем утверждать компонентную зрелость системы EPolarizationMT. Действительно, больше половины (69%) модулей относится к нулевому классу трудности (D^ ≤ 115), а в 5-й класс максимальной трудности (D^ > 430) попадает не свыше 10% (4,4% на самом деле). Поэтому велика вероятность, что за период разработки и отладки данного программного средства наукоёмкого моделирования (около 2-х лет) и его использования в экспериментах (около полугода) число остаточных дефектов было сокращено практически до нуля. Отметим, что согласно предлагаемым в [46] нормативам, на оба модуля, составляющих пятый класс, указывает, как на трудные, «безусловно требующие дополнительного интенсивного тестирования со стороны разработчика» также и D (табл.5.2). Такое тестирование было проведено. Оценки по метрике трудности D, вообще, в целом свидетельствуют об умеренной трудности системы, прогнозируя, в частности, низкую вероятность внесения ошибок при модификациях. Во-первых, верхняя граница 0* класса, куда относится половина модулей с наименьшим оценённым значением D, это всего лишь ≈71 (против ориентира 115 – верхней границы 0 класса по D^). В-вторых, из 4-х модулей класса 5*, к которому следует относить 10-ю часть наиболее трудных в смысле D модулей, три относятся к нулевому классу D^ (они имеют относительно слабо проработанную архитектуру и, тем самым, менее прозрачны других, но, зато, невелики по размеру). Наконец, хотя система в целом выглядит несбалансированной, q =-0,92, она достигнет зрелости в следующей версии («лишние» 0,02-0,03 уберутся декомпозицией 1-2 модулей).

Относительный уровень затрат на разработку, который характеризует метрика A, в данном случае (табл.5.3), для реализации, форсированной по скорости выполнения нашими методами, на порядок выше обычного для реализаций МДО, основанных на одномерных граничных уравнениях (в [9, 24, 30, 47, 48], для которых характерно значение A = $1{,}2\cdot10^{6}$bt·sym). Однако он вдвое ниже, чем для традиционных реализаций МДО на базе двумерных

граничных уравнений, (для которых характерно значение как A = $26{,}4\cdot10^{6}$bt·sym [44]). Относительно большая величиная работы программирования для программ EPolarizationMT, и EPolarizationST, прежде всего объясняется нетривиальным интерфейсом для ввода данных о сложных системах экранов, тогда как при оценке более ранних реализаций основанный на СИУ и ГСИУ такой интерфейс отсутствовал либо не учитывался. Реализация собственно МДТ у нас поддерживается модулями с суммарной работой программирования $A_0 = 5{,}1$kHd. Модификация, обеспечивающая ускорение, сводится (рис.5.3-5.5) к замене 1 базового компонента (с объёмом разработки 0,68 Hd) на 3 других (суммарный объём 18,9 Hd). Прирост работы программирования

$$\Delta A = 3{,}47\text{, и } \Delta A / A_0 = 0{,}68. \qquad (5.4)$$

Следовательно, подобная модификация основных подсистем реализации МДО должна вызывать 5-8 дополнительных ошибок кодирования (около 9% от общего числа для всей системы) и дополнительные затраты времени на разработку, которые составят ориентировочно 70% от времени реализации непосредственно МДО. При этом для дополнительных модулей СПС характерно попадание в 4-е классы трудности по обеим метрикам трудности.

5.4 Метрики для индикации и оценки снижения затрат электроэнергии при сравнении вычислительных методов

Оценка характеристик процесса взаимодействия электромагнитного поля с проводящими экранам, которые входят в состав антенных, рефлекторных и волноведущих устройств, является типичной задачей при конструировании и усовершенствовании указанных технических систем. При этом обычно волновые числа велики, но недостаточно велики для привлечения квазигеометрических методов, так что необходимо использовать полное компьютерное моделирование процесса, причём для многочисленных вариантов значений волновых чисел и параметров

конструкции. Т.о., речь идёт о массовых расчётах.

Один из наиболее эффективных подходов к компьютерному моделированию дифракционных процессов – численное решение граничных уравнений методами дискретных особенностей (МДО), тем не менее, не является «быстрым» (число операций при 2-мерных постановках имеет порядок N3, а для 3-мерных - N6). Отсюда вытекает необходимость проводить длительные вычисления, а, тем самым, - тратить значительное количество электроэнергии. Поэтому весьма актуально создание усовершенствованных модификаций вычислительных методов дискретных особенностей, которые обеспечивали бы существенную энергоэкономию.

Резервом улучшения практических характеристик вычислительных методов является создание их параллельных модификаций, ориентированных на конкретную архитектуру вычислительной системы. О каких системах речь? Несмотря на доступность в современных условиях услуг мощнейших датацентров («облачные вычисления»), вычислительных кластеров и дополнительных устройств к компьютерам типа карточек CUDA, персональные компьютеры (ПК) остаются наиболее доступным и экономичным инструментом вычислений для колоссальной армии исследователей и конструкторов. Недаром потребление электроэнергии парком ПК устойчиво растёт – на 5% в год, несмотря на неизменное повышение энергоэкономичности новых моделей. Важнейшей практической характеристикой вычислительных методов является их быстродействие. Однако активизация возможностей ускорения, скрытых в архитектуре современных ПК приводит, естественно, к более интенсивному энергопотреблению устройствами компьютера. Нужно иметь в виду, что обычный энергорасход ПК в разы меньше того максимума, который способны поддерживать его устройства питания. Если одно и то же вычисление удаётся вдвое ускорить, а энергопотребление компьютера при таком вычислении поднимется вдвое, то существенная экономия времени ожидания результатов не будет сопровождаться никакой экономией

энергоресурсов!

К сожалению, ускорение вычислений не за счёт их сокращения («математика»), а за счёт интенсификации выполнения («архитектура») сильнее загружает аппаратуру и *увеличивает расход энергии за единицу времени*.

Поэтому гарантией сохранения (возможно, уменьшения) потребления энергии при ускоренных вычислениях является выполнение условия:

$$m_0 = K_{speed} / (P_{max} / P_{base}) \geq 1.0 \qquad (5.3)$$

где K_{speed} – коэффициент ускорения вычислений,

P_{max} – номинальная (т.е. максимальная) мощность процессора (TDP),

P_{base} – оценка снизу для мощности, потреблявшейся при выполнении базовой версии;

m_0 – грубая метрика контроля энергопотребления при ускорении вычислений.

m_1 – специальная, более тонкая метрика, полагается равной отношению суммарных затрат потребления электроэнергии процессором по стандартн. оценке Intel:

$$m_1 = E_{base} / E_{speed} \qquad (5.4)$$

5.5. Эксперименты по оцениванию затрат электроэнергии тайлинговыми и векторизованными методами

Для сравнения энергопотребления мы ограничиться контролем энергопотребления процессора, так как именно он является самым большим потребителем энергии в ПК, который использован в экспериментах. Тестирование производилось на процессоре со следующими характеристиками: intel core i5-4430, 3GHz, 4 cores, 6MB cache, 84W TDP, AVX 2.0. Для мониторинга энергопотребления при работе программы была выбрана специальная утилита Intel Power Gadget. Для получения результатов затрат электроэнергии, показания о потребляемой мощности снимались

каждую 0.1 сек периода времени, начиная с момента запуска программы и до ее завершения (не менее 10 сек в проведенных экспериментах для контрольной программы).

Intel Power Gadget – это утилита предназначенная для мониторинга потребления электроэнергии, поддерживаемая процессорами Intel core (2 поколения и старше). Данная утилита включает в себя приложение, драйвер и библиотеки для мониторинга и оценки энергопотребления процессора в реальном времени в ваттах, за счет использования счетчиков энергии в процессоре. Предназначение данной утилиты – помочь конечным пользователям, заинтересованным в более точной оценке мощности с программного уровня без использования аппаратных средств.

Тестирование проводилось для двух программ: первая, контрольная – это «обычная» последовательная реализация, рассмотренная в разделе 2, вторая – экспериментальная, ускоренная на основе взаимосвязанного распараллеливания, тайлинга и векторных регистров с использованием инструкций AVX 2.0 и FMA [85].

Таблица 5.4.

Данные об энергопотреблении контрольной и экспериментальной программ при разных параметрах

Параметры: (экранов)х(особенностей)	2x500	2x1000	10x200	5x500	5x1000	10x500	10x1000
Время, счета контрольной программы (сек)	11,95	96,22	51,898	117,97	1091,6	1006,1	8688,1
Средняя мощность (вт)	14,85	15	14,99	15,01	15,04	15,03	15,04
Затрачено (вт·ч)	0,049	0,4	0,216	0,491	4,56	4,2	36,296

Время, счета экспериментальной программы (сек)	1,892	13,470	2,390	7,150	53,313	31,950	246,8
Средняя мощность (вт)	31,47	41,51	37,53	40,63	42,54	42,7	42,76
Затрачено (вт·ч)	0,017	0,155	0,025	0,081	0,630	0,379	2,931
Коэффициент ускорения	6,32	7,14	21,71	16,50	20,48	31,49	35,20
m_0	1,12	1,28	3,88	2,95	3,67	5,63	6,30
m_1	2,98	2,58	8,67	6,10	7,24	11,08	12,38

Тестирование проводилось на задачах со временем счета более 1 секунды для экспериментальной программы, в связи с невозможностью достаточно точно измерить меньшие интервалы. За базовое значение волнового числа принималось $\kappa = 3\pi$.

В табл. 5.4 приведены результаты эксперимента при разном числе рассеивающих экранов и разном числе дискретных особенностей, приходящихся на один экран в дискретной модели псевдодифференциального граничного уравнения дифракционной задачи. Варьировались также значения волнового числа (от π до 8π). При этом результаты в строке «Экономия» в первых двух значащих цифрах не изменялись

5.6 Анализ результатов оценки затрат электроэнергии тайлинговыми и векторизованными методами

При оценках по метрике m_0 величина P_{base} из (5.3) оценивается с

помощью утилиты Intel Power Gadget. Точность измерений с помощью утилит используемого типа, вероятно, не может быть ниже 10% [32]. Поэтому при оценке метрик подраздела 5.4 вместо приведенных в табл. 5.4 значений, полученных при снятии показаний, брались в одних случаях верхние (+10%), а в других случаях – нижние (-10%) оценки. Это приводит к таким оценкам метрик:

$m_0 = 1,02$ (для 2x500), $m_0 = 5,73$ (для 10x1000);

$m_1 = 2,38$ (для 2x500), $m_1 = 10,23$ (для 10x1000).

Таким образом, зафиксированная с их помощью экономия энергии в 2.5 и, тем более, в 12 раз не оставляет сомнений в том, что экспериментальная программа обеспечивает экономию, не менее, чем в 2 раза при размерности дискретной модели $N \approx 1000$, типичной при практических вычислениях с контролируемой точностью, и не менее 10 раз при $N \approx 10000$, что также нередко требуется при моделировании дифракции электромагнитных волн на фазированных решетках.

Выводы по разделу 5

На основе опыта разработки зрелой, как показывают метрические оценки, реализации ускоренного варианта МДО обоснована гипотеза об умеренных затратах на такие реализации и в будущем. Их разработка на основе методов оптимизации работы программы с кэш-памятью и векторными регистрами по затратам требуемой работы ожидается не более трудоёмкой, чем вдвое по сравнению с реализацией базовых алгоритмов МДО. Риски, связанные с надёжностью программных реализаций не должны возрастать более чем на 10%. По мере создания таких реализаций целесообразно было бы исследовать точность полученных априорных оценок.

Получена оценка энергоэффективности, достигнутой за счёт использования на ПК ускоренного вычислительного метода, который был

разработан как модификация математического метода с ориентацией на компьютерную архитектуру. Ускоренная программа экономит энергию, по крайней мере, 2-10 раз в сравнении с программой, реализующей математический метод непосредственно (в зависимости от параметров задачи).

Результаты подтверждают гипотезу о наличии общей закономерности, которая состоит в том, что компьютерные вычисления, которые удаётся заметно ускорить только за счёт усовершенствования метода, экономят в итоге не только время, но также и потребляемую компьютерной системой энергию.

Разработанная ускоренная программа компьютерного моделирования на базе метода дискретных токов [94, 1] готова к внедрению, способствуя экономии времени счета и электроэнергии.

ВЫВОДЫ

В диссертационной работе решена актуальная научно-прикладная задача, суть которой заключается в создании для численного моделирования дифракции на персональных компьютерах (ПК) вычислительных методов, которые имеют специальную, основанную на существующих МДО, схему вычислений, которая позволяет, наряду с простым распараллеливанием этих вычислений на процессорные ядра, использовать дополнительные вычислительные мощности, скрытые в архитектуре современных ПК, без особых усилий программирования и дополнительных затрат энергии при выполнении. В процессе выполнения работы получены новые научные и практические результаты.

1. На основе анализа потребностей вариантных вичислений, особенностей компьютерного моделирования дифракционных процессов, основанного на дискретных моделях граничных интегральных и псевдодиференциальних уравнений и перспективных методов ускорения вычислений на персональных компьютерах уточнено содержание основных задач диссертационного исследования. Сделан вывод о целесообразности поиска резервов преодоления главных ограничений ресурсов ПК (количество процессорных ядер и скорость доступа к основной памяти) за счет управления зернистостью вычислений и векторизации существующих МДО.
2. Впервые разработан тайлинговый параллельный вычислительный метод 2D моделирования дифракции электромагнитных волн на проводщих экранах, который относится к класу МДТ, однако отличается от известных методов, считая параллельные, схемой вычислений, которая позволяет улучшить локальность памяти при реализации в мультиядерной системе одного или нескольких ПК, существенно уменьшая время расчёта матрицы, решения СЛАУ дискретной модели граничных уравнений и диаграмм направленности.

Она также предназначена служить шаблоном аналогичных решений для других МДО. Как показывает пример с дифракцией E-поляризованных электромагнитных волн на системе 20-ти ленточных металлических экранов (разной ориентации), разработанный метод ускоряет задачи построения матрицы дискретной модели, приближенного вычисления плотности токов на экранах и диаграммы направленности при 100 дискретных особенностях на экране почти в 4 раза на 2-ядерном ПК и примерно в 8 раз в 4-х и 6-ти ядерном.

3. Усовершенствован МДТ в направлении его векторизации, которая позволяет применить AVX команды процессора, если они в ПК обеспечены, с целью использования при реализации на ПК векторных регистров для уменьшения времени расчета матрицы, решения СЛАУ дискретной модели предельного уравнений и вычисления диаграмм направленности, причем указан прозрачный метод переноса этой векторизации на другие МДО решения дифракционных задач. Само по себе ускорение вычислений МДО способом векторизации позволяет снизить общее время решения прикладных задач МДО до 3 раз, но этот эффект не зависит от применения тайлинга, и в сочетании с ним ускорение при решении типичных задач МДО происходит в среднем примерно в 20 раз.
4. Усовершенствован метод организации параллельного компьютерного моделирования дифракционных процессов, основанный на граничных интегральных уравнениях и МДО, в направлении достижения средствами высокоуровневого программирования согласованности критериев динамичности распределения ресурсов компьютера, сохранения скорости вычислений и простоты программной реализации, что позволяет, в частности, в достаточной мере реализовать потенциал быстрого выполнения разработанных в диссертации вычислительных методов, не требуя от прикладника-реализатора решения сложных задач системного программирования. Эксперименты показывают, что

при таком методе организации компьютерного моделирования дифракции возможно небольшое снижение скорости выполнения (сравнивая его с применением для тех же целей С ++ технологии) - уменьшение всего на несколько процентов, тогда как, по метрическим оценкам, работа программирования сокращается (в сравнении с применением С ++ технологии) примерно в 2.7 раза.

5. Впервые разработан метод проверки качества ПС компьютерного моделирования по характеристике расхода электрической энергии, аналогов которого по публикациям не найдено, который имеет собственные метрики, примитивы которых получаются программными средствами и с помощью документации, позволяя без специального оборудования не только проверять отсутствие перерасхода энергии , в частности, при использовании созданных в диссертации моделирующих ПС, но и сравнивать расходы различных методов и даже оценивать достоверное уменьшение расходов лучших методов при их применении. Оценки по результатам экспериментов свидетельствуют о том, что применение разработанных в диссертации вычислительных методов для компьютерного моделирования на ПК дифракционных процессов может уменьшать затраты электрической энергии на выполнение той же вычислительной работы от 2 до 10 раз.
6. Разработанные вычислительные методы, метод организации на их основе компьютерного моделирования с учетом конкурирующих требований (динамичность использования ресурсов, высокая скорость, умеренные усилия программирования) и метод оценки качества программных реализаций по усилиям программирования и расходу электрической энергии при выполнении были внедрены в ИПМ им. А.М. Подгорного, в Институте телекоммуникаций и глобального информационного пространства, в учебном процессе Харьковского национальном университетеа имени В.Н. Каразина.
7. Разработанные вычислительные методы могут быть использованы при

всестороннем исследовании элементов антенных, волноведучщих и ретранслирующих радиоэлектронных систем, а также наличия и влияния на прочность под действием упругих волн пустот и включений цилиндрической формы в материалах механичесикх конструкций средствами компьютерного моделирования на ПК, особенно в тех случаях, когда ранее ограниченность вычислительных ресурсов ПК не позволяла проводить в полном объёме в заданный срок необходимые многовариантные вычисления.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. А. с. № 64811. Україна. Комп'ютерна програма «EMPolarization» / В. О. Міщенко, Б. В. Паточкін. – Дата реєстрації: 05.04.2016.

2. Андреев А. Е. Применение векторных инструкций в алгоритмах блочных операций линейной алгебры / А. Е. Андреев, В. А. Егунов, А. А. Насонов, А. А. Новокщенов // Известия ВолгГТУ. Серия "Актуальные проблемы управления, вычислительной техники и информатики в технических системах". Вып. 21 : межвуз. сб. науч. ст. / ВолгГТУ. – Волгоград, 2014. – № 12 (139). – С. 5-11.

3. Андрианова Е. Г. Метод оценки эффективности реализации блочного алгоритма на основе графического процессора в открытой гетерогенной системе / Е. Г. Андрианова, Д. И. Мирзоян, А. Б. Петров // Журнал радиоэлектроники. –2014. – № 3, – С. 1-15.

4. Анфиногенов А. Ю. Практическое ускорение численного решения трехмерной скалярной задачи дифракции методом дискретных особенностей. / А. Ю. Анфиногенов // Вісник ХНУ. –2007. – № 775. – С. 3–10. – (Математичне моделювання. Інформаційні технології. Автоматизовані системи управління ; вип. 7).

5. Бахвалов Н. С. Численные методы / Н. С. Бахвалов, Н. П. Жидков, Г. М. Кобельков. – Москва : Наука, 1987. – 600 с.

6. Белоцерковский С. М. Численные методы в сингулярных интегральных уравнениях / С. М. Белоцерковский, И. К. Лифанов. – М.:Наука, 1985. – 256 с.

7. Боресков А. В. Основы работы с технологией CUDA / А. В. Боресков, А. А. Харламов. – М. : ДМК Пресс, 2010. – 232 с.

8. Боровинский А. В. Компьютерное моделирование 3D дифракции в плоскопараллельной среде: опыт реализации МДО методом расширенных программ / А. В. Боровинский, А. В. Гахов, В. О. Мищенко // Вісник ХНУ. – 2009. – № 863. – С. 21–35. – (Математичне моделювання. Інформаційні технології. Автоматизовані системи управління ; вип. 12).

9. Боровинский А. В. Модули библиотеки программных реализаций

алгоритмов метода параметрических представлений граничных псевдодифференциальных уравнений Ю. В. Ганделя / А. В. Боровинский, В. О. Мищенко // Труды XVII Международного симпозиума «Методы дискретных особенностей в задачах математической физики». – Харьков ; Сумы, 2015. – С. 48–51.

10. Булыгин В. С. Гиперсингулярное интегральное уравнение задачи дифракции H-поляризованной электромагнитной волны на поверхности вращения / В. С. Булыгин // Вісник ХНУ. – 2009. – № 847. – С. 40–57. – (Математичне моделювання. Інформаційні технології. Автоматизовані системи управління ; вип. 11).

11. Вайникко Г. М. Численные методы в гиперсингулярных интегральных уравнениях и их приложения / Г. М. Вайникко, И. К. Лифанов, Л. Н. Полтавский. – Москва : Янус, 2001. – 508 с.

12. Воеводин В. В. Матрицы и вычисления / В. В. Воеводин, В. А. Кузнецов. – Москва : Наука, 1984. – 320 с.

13. Воеводин В. В. Параллельные вычисления / В. В. Воеводин, Вл. В. Воеводин. – Санкт-Петербург : БХВ–Петербург, 2002. – 608 с.

14. Гандель Ю. В. О приложении идей метода дискретных вихрей к задачам электродинамики / Ю. В. Гандель, И. К. Лифанов // Научно-методические материалы по численным методам Военно-воздушной инженерной академии им. проф. Н. Е. Жуковского. – Москва, 1985 – С. 3–13.

15. Гандель Ю. В. Параметрические представления сингулярных интегральных преобразований и краевые задачи математической физики / Ю. В. Гандель // Нелинейные краевые задачи математической физики и их приложения : [сб. науч. трудов]. - Киев, 1995. – С. 65–66.

16. Гандель Ю. В. Краевые задачи для уравнений Гельмгольца и их дискретные математические модели / Ю. В. Гандель // Современная математика. Фундаментальные направления. – М., 2010. – Т. 36. – С. 36–49.

17. Гандель Ю. В. К обоснованию метода дискретных особенностей в двумерных задачах дифракции / Ю. В. Гандель, И. К. Лифанов,

Т. С. Полянская // Дифференциальные уравнения. – 1995. – Т. 31, № 9. – С. 1536–1541.

18. Гандель Ю. В. Парные и гиперсингуляные интегральные уравнения задач дифракции электромагнитных волн на плоских решетках и экранах / Ю. В. Гандель // Труды XI Международного симпозиума «Методы дискретных особенностей в задачах математической физики», Херсон, 11–18 июня 2003 г. – Харьков ; Херсон, 2003. – С. 53–58.

19. Гандель Ю. В. Введение в методы вычисления сингулярных и гиперсингулярных интегралов : [учеб. пособ.] / Ю. В. Гандель.– Харьков : ХНУ имени В. Н. Каразина, 2001. –92 с.

20. Гандель Ю. В. Математические вопросы метода дискретных токов : [учеб. пособ.]. Ч. 2 / Ю. В. Гандель, С. В. Ерёменко, Т. С. Полянская. – Харьков : ХГУ, 1992. – 145 с.

21. Гандель Ю. В. Псевдодифференциальные уравнения электромагнитной дифракции на плоскопараллельной структуре и их дискретная модель / Ю. В. Гандель, В. О. Мищенко // Вісник ХНУ. – 2006. – № 733. – вип. 6. – С. 58–75.

22. Гандель Ю. В. Параметрические представления псевдодифференциальных операторов и краевые задачи для дифференциальных уравнений электродинамики / Ю. В. Гандель // The Sixth International Conference on Differential and Functional Differential Equations, August 14–21, 2011. – Moscow, 2011. – P. 87–88.

23. Гандель Ю. В. Математические модели двумерных задач дифракции: сингулярные интегральные уравнения и численные методы дискретных особенностей / Ю. В. Гандель, В. Д. Душкин. – Харьков: Акад. ВВ МВД Украины, 2012. – 544с.

24. Гахов А. В. Поиск математической модели при анализе связи между видами качества расчетных программ / А. В. Гахов, В. О. Мищенко // Радиоэлектронные и компьютерные системы. – 2008. – № 6(33). – С. 214–218.

25. Гахов А. В. Схема параллельной модификации систем компьютерного

моделирования, использующих методы дискретных особенностей / А. В. Гахов, В. О. Мищенко // Радиоэлектронные и компьютерные системы. – 2010. – № 6 (47). – С. 143–147.

26. Давыдов А. Г. О возможностях новой версии программного комплекса EDEM / А. Г. Давыдов, Ю. В. Пименов // Тезисы докладов и сообщений I Международной научно-технической конференции «Физика и технические приложения волновых процессов», Самара, 10–16 сентября 2001 г. – Самара, 2001. – Т. 1. – С. 21–26.

27. Дмитриев В. И. Метод интегральных уравнений в вычислительной электродинамике / Дмитриев В. И., Захаров Е. В. – М.: МАКС Пресс, 2008. – 316 с.

28. Довгий С. А. Методы решения интегральных уравнений : Теория и приложения. / С. А. Довгий, И. К. Лифанов. – Киев : Наукова думка, 2002. – 343 с.

29. Дорофеева В. И. Применение параллельных технологий для решения задачи растекания бугра грунтовых вод под действием силы тяжести / В. И. Дорофеева, И. Е. Матвеев // Труды Международных школ-семинаров «МДОЗМФ». – Орёл, 2010. – Вып. 8. – С. 48–53.

30. Духопельников С. В. Программная система для поиска псевдособственых частот цилиндрического волновода со щелью, основанная на МДО / С. В. Духопельников, В. О. Мищенко // Вісник ХНУ. – 2009. – № 847. – вип. 11. – С. 167–173.

31. Захаров Е. В. Численный анализ дифракции электромагнитных волн на идеально проводящих экранах / Е. В. Захаров, Ю. В. Пименов. – Москва : АН СССР, ИРЭ, 1984. – 70 с.

32. Зелёная инженерия в 2-х томах. Том 1. Принципы, модели, компоненты (лекционный материал) / В.С. Харченко, Д.К. Андрейченко, С.Г. Антощук, Е.Н. Бульба, П.П. Гармидаров, С.В. Гонтовой, А.В. Горбенко, А.А. Гордеев, А.В. Дрозд, М.А. Дрозд, Ю.В. Дрозд, Е.Н. Зайцева, К.Я. Защелкин, А.О. Ивасюк, В.В. Калиниченко, А.Н. Каменских, А.Ю. Кривцов, В.Г. Леващенко,

А.В. Мазуренко, Е.Д. Маевская, Д.А. Маевский, С.И. Милейко, В.О. Мищенко, О.Н. Одарущенко, О.В. Олещук, А.А. Орехова, А.В. Потий, А.А. Сиора, В.В. Скляр, О.М. Тарасюк, С.Ф. Тюрин. – под. ред. Харченко В.С. – Харьков: Нац. аэрокосм. ун-т "ХАИ".– 2014.- 594 с.

33. Киркоров Л. С. Параллельные алгоритмы математических моделей: исследование локальности и применение языка Ада / Л. С. Киркоров, С. И. Киркорова // Вісник ХНУ. – 2009. – № 863. – С. 129–142. – (Математичне моделювання. Інформаційні технології. Автоматизовані системи управління ; вип. 12).

34. Колтон Д. Методы интегральных уравнений в теории рассеяния / Д. Колтон, Р. Кресс [пер. с англ. Ю. А. Еремин, Е. В. Захаров; под ред. А. Г. Свешникова] – Москва : Мир, 1987. – 311 с.

35. Корнеев В. Д. Параллельное программирование в MPI / В. Д. Корнеев. – [2-е изд.] – Новосибирск : ИВМиМГ ИСО РАН, 2002. – 215 с.

36. Лифанов И. К. О методе дискретных вихрей / И. К. Лифанов // Прикладная математика и механика. – 1979. – Т. 43, №1 – С. 184–188.

37. Лифанов И. К. Метод сингулярных интегральных уравнений и численный эксперимент / И. К. Лифанов. – Москва : Янус, 1995. – 520 с.

38. Лифанов И. К. Гиперсингулярные интегральные уравнения и теория проволочных антенн / И. К. Лифанов, А. С. Ненашев // Дифференциальные уравнения. – 2005. – Т. 41, № 1. – С. 121–137.

39. Лиходед Н. А. Методы распараллеливания гнезд циклов : [курс лекций] / Н. А. Лиходед. – Минск : БГУ, 2008. –100 с.

40. Лиходед Н. А. О выборе зерна вычислений при реализации алгоритмов на параллельных компьютерах с распределенной памятью / Н. А. Лиходед, А. К. Пашкович // Весці НАН Беларусі. – 2008. – № 2. – С. 121–123. – (Сер. фіз.-мат. навук).

41. Ложкин А. М. Взаимодействие P- и SV- волн с периодической системой цилиндрических включений в пространстве / А. М. Ложкин, А. М. Назаренко // Науково-технічна конференція викладачів, співробітників

і студентів механіко-математичного факультету, 14–29 квітня : програма і тези доп. – Суми, 2004. – С. 106–107.

42. Ложкін О. М. Дифракція пружних хвиль на періодичних системах циліндричних порожнин та жорстких включень / О. М. Ложкін, О. М. Назаренко // Акустичний вісник. – 2006. – Т. 9, № 4. – С. 35-42.

43. Медведик М. Ю. Параллельный алгоритм расчёта поверхностных токов в электромагнитной задаче дифракции на экране / М. Ю. Медведик, Ю. Г. Смирнов, С. И. Соболев // Вычислительные методы программирования. – 2005. – Т. 6. – С. 99–108.

44. Мищенко В. О. Гибкая модель приближенных вычислений ядер двумерных гиперсингулярных операторов и архитектура программной реализации / В. О. Мищенко // Вісник ХНУ. – 2008. – № 809. – вип. 9. – С. 132–147.

45. Мищенко В. О. Оптимизация компактной схемы Гаусса для многоядерных процессоров / В. О. Мищенко, Б. В. Паточкин // Вісник ХНУ. – 2011. – № 987. – вип. 18. – С. 70–81.

46. Мищенко В. О. Метрики трудности в оценке надёжности инструментальных библитек и фреймворков / В. О. Мищенко // Вісник ХНУ. – 2014. – № 1131. – вип. 25. – С. 126-147.

47. Мищенко В. О. Программное обеспечение МДО: роль математических моделей надежности и трудоемкости / В. О. Мищенко // Труды Международных школ-семинаров «МДОЗМФ». - Орел, 2005. – Вып. 4. – С. 73–80.

48. Мищенко В. О. Разработка наукоёмких моделирующих программных систем как объект моделирования / В. О. Мищенко // Современные проблемы математики и её приложения в естественных науках и информационных технологиях : тез. докл. междунар. конф., посвящ. 50-летию мех.-матем. ф-та, 17–22 апр. 2011 г. – Харьков, 2011. – С. 184.

49. Мищенко В. О. Распараллеливание счета в численном моделировании МДО дифракции волн Е-типа на проводящих экранах / В. О. Мищенко,

Б. В. Паточкин // Международная школа-конференция «Современные проблемы математики, механики и информатики»: тезисы докладов. – Харьков, 2013. – С. 106–107.

50. Мищенко В. О. Энергетический анализ программного обеспечения с примерами реализации для Ада-программ / В. О. Мищенко. – Харьков : ХНУ имени В. Н. Каразина, 2007. – 129 с.

51. Мищенко В. О. Организация и оптимизация компьютерного моделирования дифракционных процессов методами дискретных особенностей / В. О. Мищенко, Б. В. Паточкин // Вестник НТУ «ХПИ» : сб. науч. тр. – 2015. – № 41(1150). – Темат. вып. : Математическое моделирование в технике и технологиях. – С. 76–85. – Входить до науково-метричної бази «Ulrichs Periodicals Directory» (New Jersey, USA).

52. Мищенко В. О. Вычислительные эксперименты по точности моделирования МДО дифракции на идеально проводящих лентах / В. О. Мищенко, Б. В. Паточкин, А. В. Боровинский // Труды научно-технической конференции с международным участием «Компьютерное моделирование в наукоемких технологиях». – Харьков, 2014. – С. 45–48.

53. Мищенко В. О. Преимущества, затраты и риски модификации реализаций методов дискретных особенностей с целью оптимизации / В. О. Мищенко, Б. В. Паточкин // Вісник ХНУ. –2015. – Вип. 28. – С. 69–76.

54. Морева В. С. Параллельный программный комплекс POLARA для исследования расчетных схем метода вихревых элементов / В. С. Морева, И. К. Марчевский // Труды XV Международного симпозиума «Методы дискретных особенностей в задачах математической физики» (МДОЗМФ-2011). – Харьков ; Херсон, 2011. – С. 280-283.

55. Назаренко А. М. Вычислительные методы в задачах дифракции упругих волн на системах неоднородностей на базе сингулярных интегральных уравнений. / А. М. Назаренко. – Сумы : СумГУ, 2015. – 220 с.

56. Назаренко А. М. Математическое и компьютерное моделирование в задачах дифракции P- и SV- волн на системе полостей и включений /

А. М. Назаренко // Вісник ХНУ. – 2015. – Вип. 28. – С. 102–122.

57. Назарчук З. Т. Численное исследование дифракции волн на цилиндрических структурах / З. Т. Назарчук. – Киев : Наукова думка, 1989. – 256 с.

58. Носич А. А. МДО в двумерных моделях зеркальных антенн / А. А. Носич // Радіофізика і радіоастрономія. – Харків, 2004. – Т. 9, № 3. – С. 284–292.

59. Носич А. А. МДО в двумерных задачах моделирования квазиоптических антенн / А. А. Носич // Вісник ХНУ імені В. Н. Каразіна. – 2005. – № 661. – вип. 4. – С. 201–208.

60. Носич А. А. Дискретная модель открытой квазиоптической волноведущей системы на базе программного обеспечения МДО / А. А. Носич // Вісник ХНУ. – Харків, 2007. – № 780. – вип. 8. – С. 163–173.

61. Оленев Н. Н. Основы параллельного программирования в системе MPI. / Н. Н. Оленев. – Москва : ВЦ РАН, 2005. – 80 с.

62. Ортега Дж. Введение в параллельные и векторные методы решения линейных систем : [пер. с англ.] / Дж. Ортега. – Москва : Мир, 1991. – 367 с.

63. Оценка качества и экспертиза программного обеспечения. Лекционый материал / [под. ред. В. С. Харченко] – Харьков : НАУ ім. Н. Є. Жуковського „ХАІ”, 2008. – 204 с.

64. Паточкін Б. В. Модифікація методу дискретних токів для використання векторних регістрів процесору ПК при числовому моделюванні дифракції на екранах / Б. В. Паточкін, В. О. Міщенко // Сучасні проблеми моделювання та обчислювальних методів : матеріали конф., 19–22 лют. 2015 р.– Рівне, 2015. – С. 130.

65. Паточкин Б. В. Согласование тайлинга с сокетным обменом на примере решения СЛАУ в вычислительном кластере / Б. В. Паточкин // Труды научно-технической конференции с международным участием «Компьютерное моделирование в наукоемких технологиях». – Харьков, 2012. – С. 334–337.

66. Паточкин Б. В. Минимизация времени компьютерного моделирования задачи дифракции на экранах методом оптимизации локальности памяти / Б. В. Паточкин // Вісник КрНУ імені Михайла Остроградського. – 2014. – Вип. 6 (89). – С. 58–64.

67. Паточкин Б. В. Ускорение вычислений МДО за счет более полного использования особенностей архитектуры компьютера / Б. В. Паточкин // Труды XVII международного симпозиума «Методы дискретных особенностей в задачах математической физики». – Харьков ; Сумы, 2015. – С. 208–211.

68. Паточкин Б. В. Энергосберегающие аспекты ускоряющего метода компьютерного моделирования дифракции электромагнитных волн / Б. В. Паточкин // Труды международной научно-технической конференции «Компьютерное моделирование в наукоемких технологиях». – Харьков, 2016. – С. 257–259.

69. Перепёлкин Е. Е. Вычисления на графических процессорах (GPU) в задачах математической и теоретической физики / Е. Е. Перепёлкин, Б. И. Садовников, Н. Г. Иноземцева. – Москва : URSS, 2014. – 176 с.

70. Петров Д. Ю. Исследование сходимости квадратурных формул математической модели трехмерной векторной задачи дифракции / Д. Ю. Петров // Вісник ХНУ., – Харків, 2003. – № 590. – вип. 1. – С. 193-196.

71. Программа для расчета электромагнитных полей [Электронный ресурс]. – Режим доступа: http://www.edem3d.ru. – Заглавие с экрана.

72. Совместное использование MPI и OpenMP на кластерах [Электронный ресурс]. – Режим доступа: http://www2.sscc.ru/Seminars/paper/2010/MPI_OpenMP_100325.pdf. – Заглавие с экрана.

73. Харченко В. С. Методы моделирования и оценки качества и надёжности программного обеспечения : учеб. пособие / В. С. Харченко, В. В. Скляр, О. М. Тарасюк. – Харьков : НАУ «ХАИ», 2004. – 159 с.

74. Шпаковский Г. И. Реализация параллельных вычислений: кластеры,

многоядерные процессоры, грид, квантовые компьютеры / Г. И. Шпаковский. – Минск : БГУ, 2010. – 155 с.

75. Barnes J. Programming in Ada 2005/ John Barnes. – Harlow : Addison-Wesley, 2006. – XX, 828 p.

76. Burns A. Concurrent and Real-Time Programming in ADA 2005 / A. Burns, A. Wellings. – Cambridge : Cambridge University Press, 2007. – 461 p.

77. Cilk plus [Electronic resource]. – Access mode : https://www.cilkplus.org/ – Title from a screen.

78. Cooperative Cache Scrubbing / J. B. Sartor, W. Heirman, S. M. Blackburn, L. Eeckhout, K. S. McKinley // PACT '14 : Proceedings of the 23rd international conference on Parallel architectures and compilation, Edmonton, AB, Canada, 24–27 August, 2014. – New York, 2014. – P. 15–26.

79. Cooperative Vector Parallelism in JavaScript: Language and Compiler Support for SIMD / I. Jibaja, P. Jensen, J. McCutchan, D. Gohman, N. Hu, M. Haghighat, S. Blackburn, K.S. McKinley // PACT '15 : Proceedings of the 2015 International Conference on Parallel Architecture and Compilation (PACT), San Francisco, October, 2015. – Washington, 2015. – P. 407–418.

80. Davydov A. Opportunities of program EDEM for development of devices of antenna techniques / A. Davydov, E. Zakharov // Antennas. – 2006. № 10. P.52–57

81. Firuziaan M. Parallel Processing via MPI & OpenMP / M. Firuziaan, O. Nommensen Linux Enterprise, 10/2002. 386 p.

82. Gahov A. V. Parallelism for diffraction processes modeling on the base of discrete singularities methods / A. V. Gahov, V. O. Mishchenko // Труды Научно-технической конференции с международным участием «Компьютерное моделирование в наукоемких технологиях» (КМНТ–2010).– Харьков, 2010. – Ч. 2 – С. 50–53.

83. Gahov A. V. The validation of the software that was developed for calculations related to the design of antennas / A. V. Gahov, V. O. Mishchenko // Радіоелектронні і комп'ютерні системи. – 2007. – № 6 (25). – С. 180–185.

84. Green computing [Electronic resource]. – Access mode :

https://itconnect.uw.edu/wares/acquiring-software-and-hardware/green-computing. – Title from a screen.

85. Intel Intrinsics Guide [Electronic resource]. – Access mode : //software.intel.com/sites/landingpage/IntrinsicsGuide/. – Title from a screen.

86. ISO/IEC 25010:2011. Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models. - Geneva, Switzerland : International Organization for Standardization, 2011. – .

87. Mishchenko V. O. Ada programming language and specifying, modeling and distributed computing for the implementation of the discrete singularities methods / V. O. Mishchenko // Proceedings of SCALNET–2004. – Kremenchug, 2004. – P. 110-112.

88. Myers G. J. The art of software testing / Glenford J. Myers ; [revised and updated by Tom Badgett and Todd Thomas ; with Corey Sandler].–2nd ed. – Hoboken : John Wiley & Sons, 2004. – 234 p.

89. Nosich A. A. Numerical analysis of quasioptical multi-reflector antennas in 2-D with the method of discrete singularities / A. A. Nosich, Y. V. Gandel // IEEE Transactions on Antennas and Propagation. – 2007. – Vol. 57, № 2. – P. 399–406.

90. Nosich A.A. Numerical analysis and synthesis of 2-D quasioptical reflectors and beam waveguides based on an integral-equation approach with Nystrom's discretization / A. A. Nosich, Y. V. Gandel, T. Magath, A. Altintas // Journal of Optical Society of America A. – 2007. – Vol. 24, No. 9. – P. 2831–2836.

91. Nesvit K. V. Discrete mathematical model of difraction on periodic pre-Cantor gratings with shield and numerical experiment / K. V. Nesvit // Вісник ХНУ імені В. Н. Каразіна. – 2012. – № 1037. – вип. 20. – С. 146–157.

92. Nesvit K. Difraction problem of scattering and propagation TM wave on prefractal impedance strips above shielded dielectric layer / K. Nesvit. – International Journal of Applied Mathematical Research. – 2014. – 3(1). – P. 7–14.

93. OpenCL programming guide / Aaftab Munshi, Benedict R Gaster, Timothy G Mattson, James Fung, Dan Ginsburg [et al]. – Upper Saddle River : Addison-

Wesley, 2012. – LIV, 603 str. : ilustr.

94. Parallel Gaussian compact scheme [Electronic resource]. – Access mode: http://www.mediascan.by/index.files/parallel_gaussian_compscheme_win32 .zip. – Title from a screen.

95. Parallel Programming in OpenMP / R. Chandra, R. Menon, L. Dagum, D. Kohr, D. Maydan, J. McDonald. – San Francisco : Morgan Kaufmann Publshers, 2000. – 230 p.

96. Patochkin B. V. Accelerating the computation of the discrete currents method by modification takes into account the architectural features of a modern PCs / V.O. Mishchenko B.V. Patochkin // ХНУ імені В. Н. Каразіна. – 2015. – № 1156. – вип. 26. – С. 129–139.

97. The open standard for parallel programming of heterogeneous systems [Electronic resource]. – Access mode : // https://www.khronos.org/opencl/. – Title from a screen.

98. Threading building blocks [Electronic resource]. – Access mode : https://www.threadingbuildingblocks.org/ – Title from a screen.

99. Xue J. On tiling as a loop transformation / J. Xue // Parallel Processing Letters. – 1997. – № 7(4). – P. 409–424.

100. Xue J. Loop Tiling for Parallelism / J. Xue. – Boston, MA ; London : Kluwer Academic, 2000. – XVI, 256 s.

101. Xue J. Enabling loop fusion and tiling for cache performance by fixing fusion-preventing data dependences / J. Xue, Q. Huang, M. Guo // 2005 International Conference on Parallel Processing (ICPP'05). – Washington ; Brussels ; Tokyo, 2005. – P. 107–115.

СПРАВКА
про впровадження результатів кандидатської дисертації
Паточкіна Бориса Вікторовича
«Моделювання дифракційних процесів на основі методів дискретних особливостей при обмежених обчислювальних ресурсах»
в Інституті проблем машинобудування ім. А.М. Підгорного НАН України

Цим засвідчуємо про те, що при виконанні робіт за бюджетною темою № 29 «Розробка комп'ютерної технології моніторингу сховищ легкозаймистих та хімічних речовин» було використано такі результати кандидатської дисертації Паточкіна Б.В.

1. Шаблон для перетворення МДО рішень дифракційних задач з метою прискорення їх виконання за рахунок тайлінгу, застосований при числовому вирішенні на персональному мультиядерному комп'ютері крайових задач для рівняння Гельмгольца і систем таких рівнянь.

2. Векторизація метода-зразка під AVX регістри та метод її перенесення на інші крайової задачі для рівняння Гельмгольца для або систем таких рівнянь.

3. Метод перевірки рівня витрат електричної енергії в процесі комп'ютерного моделювання та його метрики, примітиви яких отримуються програмними засобами та з документації.

Це дозволило організувати комп'ютерні експерименти з варіантного моделювання хвильових явищ із своєчасним розрахунком достатньої кількості варіантів, які призначені для розгляду при отриманні уточнених прогнозів відносно продовження термінів роботи нафто-хімічного обладнання.

Провідний науковий співробітник
Інституту проблем машинобудування
ім. А.М. Підгорного НАН України
доктор технічних наук,
професор **О.О.Стрельникова**

Старший науковий співробітник
Інституту проблем машинобудування
ім. А.М. Підгорного НАН України
кандидат технічних наук, **В.І. Гнитько**

Підписи В. І. Гнитька та О. О. Стрельникової
засвідчую:

03 03 2017

НАЦІОНАЛЬНА АКАДЕМІЯ НАУК УКРАЇНИ
ІНСТИТУТ ТЕЛЕКОМУНІКАЦІЙ І ГЛОБАЛЬНОГО ІНФОРМАЦІЙНОГО ПРОСТОРУ

вул.Горького, 180, м.Київ, 03150, тел/факс (044) 245-88-38, тел. 245-87-97
E-mail: itelua@kv.uktel.net

_____ № _____ На № _____ від _____

Затверджую
В.о. директора Інституту,
д.т.н., проф.,
чл.-кор. НАН України
О.М.Трофимчук
«15» 2016 р.

Акт
впровадження
наукових результатів

Цей акт складений про те, що результати кандидатської дисертації Паточкіна Бориса Вікторовича «Моделювання дифракційних процесів на основі методів дискретних особливостей при обмежених обчислювальних ресурсах», а саме:

1. Тайлінговий обчислювальний метод моделювання дифракції пружних хвиль на неоднорідностях елементів конструкцій та у матеріалах, заснований на схемі метода дискретних особливстей;

2. Метод перенесення прийомів векторизації відомого обчислювального метода дискретних струмів на інші МДО рішення дифракційних задач, застосований у комп'ютерному моделюванні на персональному комп'ютері дифракції пружних хиль на неоднорідностях;

3. Метод організації паралельного комп'ютерного моделювання дифракційних процесів, заснований на граничних інтегральних рівняннях і МДО з узгодженістю критеріїв простоти програмної реалізації, підвищення швидкості обчислень і можливості динамічного розподілу ресурсів комп'ютера, що дозволяє реалізувати потенціал швидкого виконання вказаних вище обчислювальних методів на персональному компютері без залучення кваліфікованого системного програміста

було впроваджено в Інституті телекомунікацій і глобального інформаційного простору НАН України в рамках комплексу науково-дослідних робіт, які виконуються за темою "Розробка обчислювальних технологій та методів моделювання для дослідження нестаціонарних процесів".

Члены комісії:

О.Г. Лебідь,
к.т.н., заст директора

О.А. Гуржій
д.ф.-м.н., пров.н.с.

Д.І. Черній,
к.ф.-м.н., пров.н.с.

З А Т В Е Р Д Ж У Ю
Проректор з науково-педагогічної
роботи Харківського
національного університету
імені В. Н. Каразіна,
академік НАН України, професор
М. О. Азарєнков
_____ 2016 р.

АКТ

Про використання результатів
дисертаційної роботи **Паточкіна Бориса Вікторовича**
«Моделювання дифракційних процесів на основі методів дискретних особливостей при обмежених обчислювальних ресурсах»

СКЛАД КОМІСІЇ:

Лазурик Валентин Тимофійович, декан факультету комп'ютерних наук, д.ф.-м.н., професор;

Міщенко Віктор Олегович, в.о. завідувача кафедри моделювання систем і технологій, д.т.н., професор;

Стєрвоєдов Микола Григорович, завідуючий кафедри електроніки та систем управління, к.т.н., доцент.

Цей акт складено в тому, що теоретичні та практичні результати дисертації Паточкіна Б. В., отримані ним відповідно до теми «Математичне моделювання фізичних процесів і чисельний експеримент» (№ДР 0104U0002366) і «Методи дозиметрії електронного випромінювання зі застосуванням комп'ютерного моделювання» (ДР 0109U001436) впроваджені в навчальний процес, а саме:

1. Тайлінговий паралельний обчислювальний метод 2D моделювання дифракції електромагнітних хвиль на провідних екранах, який істотно зменшує час цього моделювання, застосовуються в лабораторних роботах з дисципліни практичної підготовки бакалаврів «Технологія розподілених систем та паралельних обчислень» на песональних компютерах у класах та у обчислювальному міні-кластері факультету.
2. Метод векторизації дифракційних обчислень, орієнтований на використання AVX команд процесора персонального комп'ютера, демонструється в якості приклада векторних обчислень на практичних заняттях з дисципліни фундаментальної підготовки бакалаврів комп'ютерних наук «Чисельні методи».
3. Метод контролю комп'ютерного моделювання за характеристикою витрат електричної енергії на основі нових метрик дисертації, примітиви яких оцінюються програмними засобами, що дозволяє без спеціального обладнання контролювати витрати і отримувати економію енергії при виконанні великих за об'ємом робіт за допомогою наукоємних моделюючих і управляючих систем, застосовується у дисципліні за вибором студента «Системи автоматичного контролю і управління» у курсі підготовки бакалаврів ФКН.
4. Метод організації паралельного комп'ютерного моделювання дифракційних процесів на основі засобів високорівневого програмування з узгодженням критеріїв динамічності розподілу ресурсів комп'ютера, збереження швидкості обчислень та простоти програмної реалізації застосовується в лекційному курсі та на лабораторних роботах з дисципліни «Масштабованиі програмні системи» в курсі підготовки магістрів спеціальності «Інформаційні управляючі системи та технології».

Голова комісії **В. Т. Лазурик**

Член комісії **В. О. Міщенко**

Член комісії **М. Г. Стєрвоєдов**