НАНОСИСТЕМЫ: ФИЗИКА, ХИМИЯ, МАТЕМАТИКА, 2012, 3 (5), С. 42–61

УДК 519.63-37-73:535+537.8

МОДЕЛИРОВАНИЕ ИНТЕГРАЛЬНЫХ СХЕМ НАНОФОТОНИКИ: МЕТОД FDTD

К. С. Ладутенко[1,2,*], П. А. Белов[1]

[1] Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, Санкт-Петербург, Россия

[2] Федеральное государственное бюджетное учреждение науки Физико-технический институт им. А.Ф. Иоффе Российской академии наук, Санкт-Петербург, Россия

*fisik2000@mail.ru

PACS 07.05.Tp, 42.15.Eq, 42.79.-e

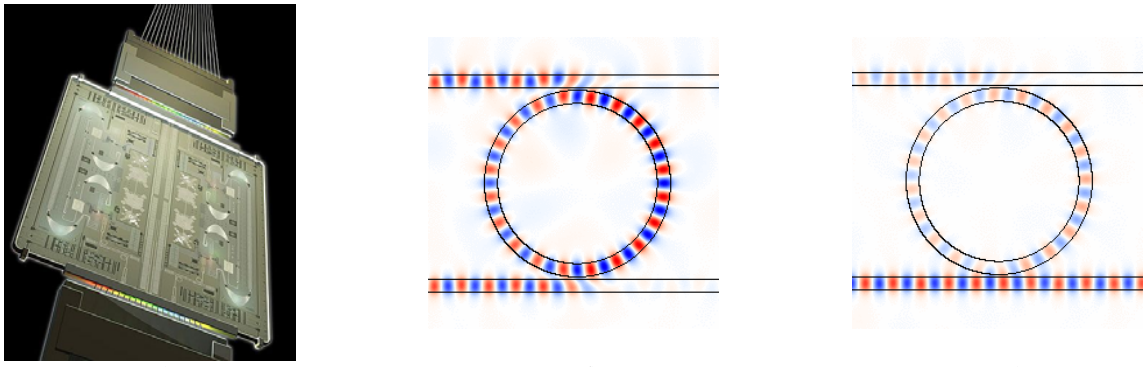
В работе рассматриваются возможности метода FDTD (метода конечных разностей во временной области) для параллельных вычислений, перечислены основные сложности, возникающие при его практическом применении, проведено сравнение с другими численными методами моделирования электромагнитных явлений. Сформулированы требования к балансу характеристик узла суперкомпьютера, при которых возможна эффективная реализация параллельной версии метода FDTD. Показана необходимость проведения вычислений экзафлопсного масштаба для решения задач интегральной нанофотоники, предложен подход к решению таких задач.

Ключевые слова: FDTD, метод FDTD, параллельный FDTD, Finite Difference Time Domain, МКРВО, метод конечных разностей во временной области, моделирование.

1. Введение

Существует огромное число методов компьютерного моделирования явлений электромагнетизма, лежащих в основе моделирования отдельных компонент и интегральных схем нанофотоники. Среди них выделяется метод конечных разностей во временной области (FDTD — Finite Difference Time Domain), который совершил революцию в проектировании приборов и устройств, взаимодействующих с электромагнитным излучением. Этот метод использует дискретизацию уравнений Максвелла по конечно-разностной схеме, что позволяет уверенно использовать его как для повседневных инженерных расчётов, так и для разработки (включая фундаментальные исследования) принципиально новых приборов и их частей, таких как, например, фотонный компьютер, элементы трансформационной оптики, плазмоники.

Особенностью метода FDTD является его большая вычислительная трудоёмкость. Существующие программные продукты позволяют моделировать небольшие фрагменты интегральных схем нанофотоники (Рис. 1), потребляя при этом до нескольких сотен гигафлопс вычислительных ресурсов в течение нескольких суток. Для моделирования с достаточной точностью крупных интегральных схем нанофотоники с большим числом близко расположенных (и влияющих на работу друг друга) элементов потребность в вычислительных ресурсах многократно возрастает. На фоне стремительно растущих вычислительных ресурсов, доступных исследователям, особую актуальность приобретает вопрос эффективного использования этих ресурсов пакетами моделирования в суперкомпьютерном окружении.



а) б) в)

РИС. 1. а) Общий вид интегральной оптической схемы и результаты моделирования фрагмента схемы - оптического демультиплексора б) в режиме коммутации частоты, $\lambda = 834$ нм в) в режиме пропускания, $\lambda = 850$ нм. Материал волноводов $AlGaAs$, их ширина 150 нм, внутренний радиус кольца 853 нм, зазор между кольцевым резонатором и плоскими волноводами 20 нм

Настоящая работа посвящена вопросу масштабирования производительности метода FDTD для экзафлопс-вычислений на примере моделирования интегральных схем нанофотоники.

1.1. Метод FDTD

Метод FDTD является популярным методом численного решения задач электромагнетизма, доказавшим свою универсальность и надёжность, оставаясь, вместе с тем, относительно простым для практической реализации [1].

В основе метода лежит дискретизация уравнений Максвелла по конечно-разностной схеме [2, 3]. Пространству модели сопоставляется сетка из конечного числа регулярно расположенных узлов, в каждом узле задаётся значение одной из компонент электрического или магнитного поля. Каждая частная производная в уравнениях Максвелла заменяется отношением разности между значениями компонент поля в близко расположенных узлах к расстоянию между этими узлами (в пространстве и времени). Отличительной чертой, позволившей FDTD выделиться в отдельный метод, стало особое расположение компонент поля по узлам сетки. Предложенное в оригинальной работе Йи [4], оно естественным образом позволяет получать численные решения уравнений Максвелла для очень большого набора задач.

В рамках такого подхода для самого простого одномерного случая компоненты $H_y^t(x)$ магнитного и $E_z^t(x)$ электрического поля (пространственная координата x и время t для магнитного поля смещены относительно электрического на половину пространственного шага и половину шага по времени) в каждый следующий момент времени можно выразить через предыдущие значения как:

$$H_y^{n+1/2}\left(i+1/2\right) = H_y^{n-1/2}\left(i+1/2\right) + \frac{\Delta_t}{\mu\Delta_x}\left[E_z^n(i+1) - E_z^n(i)\right] \quad (1)$$

$$E_z^{n+1}(i) = E_z^n(i) + \frac{\Delta_t}{\varepsilon\Delta_x}\left[H_y^{n+1/2}\left(i+1/2\right) - H_y^{n+1/2}\left(i-1/2\right)\right] \quad (2)$$

Индексы n и i — нумеруют пространственный шаг и шаг по времени для электрического поля, Δ_t и Δ_x — величины этих шагов, ε и μ — относительные диэлектрическая и магнитная

проницаемости, $E_z^n(i)$ — величина z компоненты электрического поля в момент времени $t = n\Delta_t$ с пространственной координатой $x = i\Delta_x$, остальные компоненты полей записаны аналогичным образом.

По мере своего развития метод FDTD столкнулся с целым рядом сложностей. Прежде всего хотелось бы отметить вопрос, связанный с границей модели. Метод FDTD производит вычисления полей в каждой точке моделируемого объёма, в связи с чем этот объём принципиально ограничен количеством доступной (супер)компьютеру оперативной памяти. Наличие границ модели может существенно влиять на результаты моделирования, вследствие паразитного отражения от границ падающих на них электромагнитных волн. Несколько основных направлений решения этой проблемы:

(1) Использование периодических граничных условий.
(2) Использование отражающих граничных условий: размещение вдоль границы идеальной электрической или магнитной стенки.
(3) Размещение вдоль границ слоя материала с высоким показателем поглощения — absorbing boundary condition (ABC) [5]. Существенные проблемы возникают при использовании ABC для дисперсных материалов. По мере развития ABC стали отражать существенно меньше (см., например, HABC [6] и CCOM [7]).
(4) Размещение вдоль границ слоя гипотетического материала, специально устроенного таким образом, чтобы полностью поглощать падающее на него излучение. Такой вариант называют perfectly matching layer (PML) [8]. В одном из вариантов, более простом для понимания, представляет из себя анизотропно поглощающий слой (UPML) [9], в другом (более универсальном) позволяет включать в состав границы произвольный (например, нелинейный и дисперсный) материал (CPML) [10]. В целом возможна реализация PML для большинства возможных ситуаций, при этом относительная ошибка оказывается на несколько порядков величины меньше по сравнению с оригинальным ABC при прочих равных. Были предприняты попытки экспериментальной реализации PML [11].

Дискретизация всего пространства модели регулярной сеткой определяет сильные и слабые стороны метода FDTD. К сильным сторонам относится возможность естественным образом описывать в модели взаимодействие электромагнитного поля со «сложными» материалами: анизотропными, нелинейными, дисперсными и так далее. Есть варианты алгоритма, позволяющие моделировать проводники и активные среды (усиливающие проходящую через них волну). Естественным образом реализована возможность моделирования материалов, чьи параметры непрерывно меняются в зависимости от координаты. Есть, безусловно, и слабые стороны.

Во-первых, это ступенчатая аппроксимация гладких изогнутых поверхностей, возникающая в результате дискретизации модели прямоугольной сеткой. Для борьбы со ступенчатой аппроксимацией используется целый набор модификаций метода FDTD [1], изменяющий либо начальное задание параметров модели (таких как ε и μ), либо уже сами итерационные уравнения.

Во-вторых, это вычислительная трудоёмкость метода, которая становится особо критичной для объектов, включающих в себя особенности на разных пространственных масштабах. Для того чтобы правильно учесть вклад «мелких» особенностей, шаг дискретизации должен быть значительно меньше их характерного масштаба. Так как метод изначально приспособлен только к разбиению с постоянным шагом, то общее число узлов сетки разбиения становится очень большим.

Преодоление указанной сложности возможно двумя способами:

(1) Совершенствованием используемого алгоритма дискретизации уравнений Максвелла. В настоящее время для областей с «мелкими» особенностями были предложены [1] и успешно используются различные варианты с
   - более плотной, чем для остального объёма модели, сеткой,
   - конформными преобразованиями сетки,
   - применением метода конечных элементов,
   - применением специальных уравнений, когда «мелкая» особенность — это тонкий металлический слой.

   Подобные приёмы позволяют в несколько раз сократить время вычислений при сохранении сходимости и достоверности метода, хотя, например, применение метода конечных элементов затрудняет полноценное использование всех сильных сторон метода FDTD.

(2) Реализация в параллельном и хорошо масштабируемом варианте базовых алгоритмов метода FDTD. В этом случае для уменьшения времени вычислений экономически целесообразными могут оказаться инвестиции в увеличение вычислительной мощности аппаратной базы (приобретение более мощного компьютера, наращивание вычислительных мощностей существующего кластера), а не в совершенствование алгоритма. Вызвано это тем, что (согласно закону Мура [12]) вычислительные мощности, доступные за одну и ту же сумму денег, удваиваются каждые полтора года, а расходы на разработку, реализацию и валидацию нового алгоритма не меняются или даже растут (вместе с ростом сложности алгоритма). Учитывая текущее (весьма зрелое) состояние метода FDTD, возможность увеличения эффективности последовательного алгоритма в два раза каждые полтора года представляется крайне маловероятной.

Таким образом, современная и универсальная реализация метода FDTD должна, при относительной простоте алгоритмов, применяемых для расчётов , максимально эффективно использовать ресурсы в параллельном (суперкомпьютерном) окружении.

1.2. Сравнение с другими численными методами

Кроме метода FDTD существует огромное число методов компьютерного моделирования явлений электромагнетизма, сравнительный анализ части которых произведён в [13] [14] [15] [16] [17]:

- метод конечных элементов (МКЭ, finite element method, FEM)
- метод конечных объёмов во временной области (finite volume time-domain, FVTD)
- метод моментов (method of moments, MoM), как правило реализуемый в рамках метода граничных элементов (boundary element method, BEM)
- метод конечных интегралов (finite integration technique, FIT)
- метод конечных разностей в частотной области (finite difference frequency domain, FDFD)
- псевдоспектральный метод во временной области (pseudospectral time domain method, PSTD)
- метод матриц линий передач (transmission line matrix method, TLM)

Здесь не упоминаются модификации и усовершенствования этих методов (иногда существенным образом меняющие исходный алгоритм), как и не упоминается большое число других методов. В целом, каждый из методов можно пытаться классифицировать по следующим параметрам: метод основан на интегральной или дифференциальной форме

уравнений Максвелла, метод оперирует данными во временной или в частотной области, дискретизации подвергается вся модель или только границы её составных объёмов и т.д.

Сразу стоит отметить относительную (по сравнению с более универсальными [18] методом конечных элементов и методом конечных объёмов) простоту и надёжность FDTD для случаев, когда необходимо учесть анизотропию и дисперсию материалов в сложной пространственной геометрии моделируемого объекта, а также для случаев, когда параметры материалов в модели непрерывно меняются с координатой.

Сравнение метода FDTD с другими методами приводится во многих источниках. В [14] перечисляются такие достоинства метода FDTD как малое время разработки работоспособной программы, простота метода для понимания и то, что метод работает с уравнениями Максвелла в явном виде, не привлекая приёмы линейной алгебры, а также его недостатки: ступенчатая аппроксимация и большая вычислительная сложность. При сравнении с методом FVTD отмечается, что последний лучше подходит для неоднородных объектов, время моделирования сопоставимо с временем метода FDTD, а основным недостатком является необходимость дискретизации объёма модели неоднородной сеткой (что в общем случае является нетривиальной задачей). Сильные стороны метода FDFD демонстрируются в случае, когда необходимо получить установившееся решение для одной частоты. Особо ярко это проявляется для материалов, чья зависимость от частоты не может быть формализована простыми моделями для метода FDTD. Достоинства FEM аналогичны достоинствам метода FVTD, а основной недостаток состоит в том, что необходимо решать всю систему уравнений (она может быть очень большой) для всего объекта моделирования сразу. PSTD, относящийся к спектральным методам, характеризуется тем, что использует разложение (чаще всего Фурье) полей общего решения модели. При этом используется значительно менее плотная сетка дискретизации, что даёт существенный выигрыш в задействованных памяти и вычислительных ресурсах компьютера.

В книге [16] для выбранного пространственного размера задачи (3D) приводится вычислительная сложность разных методов в зависимости от частоты f изучаемого электромагнитного поля. Для FDTD число операций растёт как $O(f^4)$, основной недостаток — ступенчатая аппроксимация границ, проходящих под углом к направлениям прямоугольной сетки дискретизации. FVTD хорошо справляется со сложными геометриями объектов модели, имеет ту же сложность, что и FDTD, но обладает слабой «отложенной» нестабильностью. Вычислительная сложность FEM растёт как $O(f^4)$ и для частотной, и для временной области, он более стабилен, чем FVTD. Для регулярной 3D сетки дискретизации TLM может быть представлен в форме, эквивалентной FDTD. FIT обладает вычислительной сложностью FDTD, но позволяет использовать произвольные сетки дискретизации с сохранением стабильности. Вычислительная сложность MoM зависит от выбранного метода решения системы уравнений. Для fast multipole method (FMM) это $O(f^3)$, а для multilevel fast multipole algorithm (MLFMA) это $O(f^2 \lg f)$.

В книге [17] на одной и той же аппаратной платформе производилось моделирование общего набора задач с помощью коммерчески доступных пакетов, основанных на разных (указанных в скобках) методах: HFSS (FEM), CST (FIT), GEMS (FDTD), FEKO (MoM). Сравнение результатов расчётов даёт довольно хорошее совпадение для CST и GEMS, которые оказались способны решить весь набор тестовых задач. GEMS оказался быстрее (иногда в несколько раз) CST и использовал меньшее количество оперативной памяти.

1.3. Параллельный метод FDTD

Весомым преимуществом метода FDTD является его высокий потенциал к распараллеливанию при декомпозиции объёма модели между вычислительными процессами,

обусловленный сильной локальностью вычислений. Последнее напрямую связано с тем, что в методе FDTD дискретизируются записанные в дифференциальном виде уравнения Максвелла, которые далее решаются итерационно по времени. Однако существующее состояние дел в области эффективного распараллеливания алгоритма FDTD оставляет желать лучшего. Так, например, Meep [19] (один из наиболее законченных FDTD пакетов моделирования с открытым исходным кодом) при увеличении используемых для вычислений процессоров с 2 до 16 ускоряется менее чем в 2 раза (сеть — Infiniband), т.е. его использование на кластерах более 16 процессоров бессмысленно. В существующих коммерческих пакетах ситуация обстоит немногим лучше. Acceleware [20] обещает ускорение не более 50 раз (GPU), GEMS [21] предлагает решение максимум для 100 процессоров (в литературе [13] упоминается масштабирование GEMS до 4000 процессоров), OmniSim [22] не позволяет использовать кластер из более 60 компьютеров, XF7 [23] кластер из нескольких компьютеров и т.д., в то время как существующие суперкомпьютерные кластеры [24] объединяют десятки тысяч процессорных узлов, в одном компьютере может быть более миллиона вычислительных ядер.

Более того, зачастую оказывается, что даже в случае возможности запуска метода FDTD на неограниченном числе узлов время, необходимое для выполнения расчёта, растёт с числом используемых узлов (вместо того, чтобы уменьшаться). Для того чтобы понять причины, лежащие в основе такого поведения, и то, как с ним бороться, необходимо рассмотреть поведение обобщённого параллельного алгоритма.

Производительность параллельного алгоритма зависит от большого числа факторов. В качестве основных причин, которые мешают достигнуть идеальной эффективности, можно назвать следующие причины общего плана:

(1) Наличие последовательных вычислений в алгоритме.
(2) Конкуренция между узлами за общие ресурсы.
(3) Накладные расходы на коммуникации между узлами.

1.3.1. Влияние на производительность последовательных вычислений. В любом алгоритме, который планируется выполнять параллельно, можно условно выделить некую долю вычислений α, которая может быть выполнена только последовательным образом. Соответственно время T_α, затрачиваемое на выполнение этих последовательных вычислений, не зависит от числа задействованных в работе алгоритма узлов p. Оставшаяся доля вычислений $1-\alpha$ может быть распараллелена идеально, т.е. $T_{1-\alpha}(p) = T_{1-\alpha}(1)/p$, где $T_{1-\alpha}(1)$ — время, затрачиваемое одним узлом на выполнение параллельной части алгоритма, а $T_{1-\alpha}(p)$ — время, затрачиваемое на выполнение этой же части алгоритма при использовании p узлов. Общее время выполнения вычислений $T_{\text{total}}(p) = T_\alpha + T_{1-\alpha}(p)$, а ускорение в результате использования p узлов определяется как

$$S(p) = \frac{T_{\text{total}}(1)}{T_{\text{total}}(p)} = \frac{T_{\text{total}}(1)}{T_\alpha + T_{1-\alpha}(p)} = \frac{T_{\text{total}}(1)}{T_\alpha + \dfrac{T_{1-\alpha}(1)}{p}} \tag{3}$$

Учитывая, что $T_\alpha/T_{\text{total}}(1) = \alpha$, а $T_{1-\alpha}(1)/T_{\text{total}}(1) = 1-\alpha$, получаем выражение, известное как закон Амдала [25]:

$$S(p) = \frac{1}{\alpha + \dfrac{1-\alpha}{p}}$$

Из него следует, что если $\alpha \neq 0$, то вне зависимости от числа используемых процессов ($p \to \infty$) общее время выполнения вычислений нельзя уменьшить более чем в $1/\alpha$ раз. В

частности, если алгоритм содержит 1% последовательных вычислений, то для 100 узлов ускорение составит около 50 раз, а для 1000 узлов — около 90 раз.

К счастью для метода FDTD, доля последовательных вычислений в нём очень мала. В наиболее трудоёмких частях вычислений (например, итеративное вычисление значений электрических и магнитных полей, начальное задание используемых в вычислениях констант) последовательные операции отсутствуют. К безусловно последовательным операциям можно отнести декомпозицию общего объёма модели на части, каждая из которых в дальнейшем будет обрабатываться отдельным узлом. Однако такие операции занимают относительно мало времени и в целом, как правило, не являются определяющими при моделировании методом FDTD.

1.3.2. Конкуренция между узлами за общие ресурсы. Ещё одной причиной, которая может существенно снизить производительность параллельного алгоритма, является конкуренция между узлами, производящими вычисления, за общие ресурсы. Так как центральный процессор является наиболее быстрой частью узла суперкомпьютера, то подобную конкуренцию можно наблюдать достаточно часто (особенно если присутствует несколько процессоров в узле, и они являются многоядерным). Среди наиболее распространённых примеров можно перечислить конкуренцию за:

- Доступ к оперативной памяти.
- Ввод-вывод в подсистеме долговременной памяти.
- Использование сетевых интерфейсов.

Коренными причинами конкуренции за общие ресурсы можно считать экономические. В настоящее время почти любую ситуацию с конкуренцией в узле за ресурсы можно разрешить, однако стоимость одного такого узла (при той же номинальной производительности) заметно возрастёт. В то же время, заметный прирост практически достижимой производительности будет наблюдаться только в узком кругу задач. Таким образом, только в случае, когда планируется использовать компьютер только для такого круга задач, подобная специализация узла становится экономически оправданной.

Для задач, требующих высокой пропускной способности оперативной памяти, возможно использование высокочастотных низколатентных модулей памяти в многоканальном режиме. Удельная стоимость за единицу оперативной памяти в подобных модулях в несколько раз больше, чем в модулях, производимых для массового рынка. Следующим фактором, определяющим возможность эффективной работы с подсистемой памяти, является размер кэша в самом процессоре и размер регистровой памяти, доступный одному ядру. Заметно больший размер кэша, улучшенные алгоритмы работы с ним, большее число регистров — одни из причин, определяющих высокую стоимость серверных процессоров.

Подсистема долговременной памяти является одной из самых медленных в компьютерах, что вызвано, прежде всего, использованием механического перемещения считывающей/записывающей головки по поверхности вращающегося магнитного диска (HDD). Твердотельные накопители, напрямую подключаемые к шине PCIe, обладают в несколько раз (а иногда и на несколько порядков) превосходящими HDD характеристиками, впрочем, это касается и их цены.

Работа сетевых интерфейсов является определяющей, когда речь заходит о масштабировании производительности суперкомпьютера в целом. Среди основных сценариев возникновения конкуренции между вычислителями можно назвать три:

(1) Конкуренция между ядрами процессора/процессоров узла за использование его единственного сетевого интерфейса.

(2) Конкуренция между несколькими узлами за ресурсы коммутатора в некоторых сетевых топологиях (например, «звезда» и «дерево»).
(3) Конкуренция между несколькими узлами за доступ к одному узлу (возникающая в случае, если такой узел управляет работой остальных узлов или, например, содержит плату специализированного ускорителя вычислений).

Большинство таких ситуаций можно решить вместе с увеличением стоимости сетевой инфраструктуры: установкой достаточного числа сетевых интерфейсов на каждый узел, использованием более сложных сетевых топологий (например, 6D сетка/тор).

В случае наличия узлов, оборудованных специальными платами расширения (выполняющих, например, вычисления аппаратным образом), для устранения конкуренции за использование такого ресурса необходимо увеличивать их число. И только в случае, если особая роль какого-то узла вызвана алгоритмическими причинами, требуется пересмотр общего алгоритма с целью уменьшения роли такого узла или полного отказа от его использования в такой роли.

В целом можно отметить, что для получения максимальной (а иногда просто приемлемой) производительности, как правило, необходимы капитальные вложения в используемый вычислительный комплекс. Однако учёт хорошо известных слабых сторон бюджетных систем при разработке и реализации алгоритма вычислений может привести к достижению достаточного уровня производительности и для таких систем.

1.3.3. Накладные расходы на коммуникации между узлами. При рассмотрении вопроса конкуренции за ресурсы уже отмечалась важная роль сетевых интерфейсов в обеспечении производительности и масштабируемости алгоритма вычислений. Оказывается, что даже в случае отсутствия конкуренции коммуникационная сеть, объединяющая узлы, может ограничивать производительность и масштабируемость алгоритма вычислений, а в некоторых случаях даже может приводить к их деградации.

Производители сетевого оборудования указывают, как правило, для своей продукции такие характеристики, как пропускная способность и латентность. Если узлам необходимо относительно редко обмениваться большими объёмами информации (в терминах теории распределённых вычислений — большими сообщениями), то более важной оказывается пропускная способность — количество данных, которое можно передать по сети за единицу времени. Если есть необходимость в частом обмене небольшими сообщениями, то определяющей становится латентность — время, необходимое для установления соединения между узлами сети.

Для обоих типов сообщений может возникнуть ситуация, когда узел перестаёт успевать обмениваться сообщениями. Тогда он вынужден приостанавливать выполнение вычислений и простаивать, ожидая завершения операций по коммуникации с другими узлами, что приводит к уменьшению эффективности работы программы. В предельном случае увеличение числа узлов приводит к такому уменьшению эффективности, что общее время выполнения вычислений возрастает. Другими словами, программа формально использует больше вычислительных мощностей, а работает медленнее.

Для достижения близкой к максимальной эффективности масштабирования программы естественным, в данном случае, решением будет создание алгоритма, в котором количество коммуникаций на один узел не растёт при увеличении числа используемых программой узлов. Тогда, при условии некой минимальной загруженности узла вычислениями, можно ожидать линейного роста производительности программы с ростом числа узлов.

В случае большого объёма информации, передаваемого между узлами, существенным может оказаться ещё один фактор: для обмена сообщениями тратится часть вычислительных ресурсов узла. Из практического опыта следует, что, например, при использовании сети Ethernet для обеспечения работоспособности стека TCP/IP на скорости 1 Гбит используется около 1 ГГц производительности одного вычислительного ядра. Для алгоритмов, требовательных к пропускной способности коммуникационной сети, желательно использовать коммуникационную сеть, создающую минимальную дополнительную нагрузку на вычислитель (такую как, например, InfiniBand, NUMAlink и т.д.)

1.3.4. Общие критерии масштабируемости. Само по себе определение масштабируемости может варьироваться в зависимости от решаемой задачи. Один из подходов сформулирован в предположениях закона Амдала [25] (см. раздел 1.3.1). В нём для одной и той же задачи моделируется ускорение при увеличении числа узлов, выполняющих расчёт, а фактором, ограничивающим ускорение, является наличие в алгоритме некой доли α последовательных вычислений. Так как общее время вычислений T_{total} не может быть меньше времени T_α, затрачиваемого на выполнение последовательных вычислений, то возможности по ускорению программы оказываются существенно ограничены, если соотношение $T_{\text{total}} \gg T_\alpha$ перестаёт выполняться.

На практике больший интерес представляет не абстрактное значение относительного ускорения работы вычислительной программы, а абсолютное значение времени T_{user}, которое пользователь подобной программы проводит в ожидании с момента ввода исходных данных до момента получения результата. Если при пропорциональном изменении размера задачи моделирования и числа используемых узлов время T_{user} практически не меняется, то для пользователя это означает идеальную масштабируемость. Такой подход, предложенный Густавсоном [26], накладывает значительно менее строгие ограничения на последовательную часть алгоритма вычислений. А именно, для идеальной масштабируемости он требует всего лишь независимости количества последовательных вычислений от размера задачи, а не полного их отсутствия.

Интерес представляет и некий смешанный случай, когда доля последовательных вычислений α мала и медленно растёт с числом узлов. В такой ситуации масштабирование (в рамках подхода Густавсона) будет оставаться почти идеальным до тех пор, пока долей α можно пренебречь. Именно такая ситуация, как уже отмечалось в конце раздела 1.3.1, характерна для метода FDTD.

В методе FDTD размер вычислительной задачи определяется количеством узлов сетки дискретизации, которая накладывается на моделируемый объём. Для максимальной эффективности количество оперативной памяти одного вычислительного узла должно быть сбалансировано с его производительностью и характеристиками коммуникационной сети. Это ограничивает размер задачи, которую можно эффективно решать на одном узле. Однако, например, иногда речь идёт о принципиальной возможности запуска большой задачи (и есть возможность долго ждать результата), тогда баланс смещается в пользу большего объёма оперативной памяти в ущерб производительности вычислителей и сети.

2. Предположения модели

Моделируя поведение алгоритма, пытаясь разобраться в том, что может влиять на его масштабируемость и эффективность, необходимо чётко понимать условия применимости такой модели. В рамках настоящей статьи прежде всего обсуждаются кластерные суперкомпьютерные системы. Учитывая более высокий уровень интеграции в MPP (massively

parallel processing) суперкомпьютерах, стоит ожидать, что программы, эффективно работающие на кластерах, на MPP суперкомпьютерах смогут продемонстрировать аналогичный или лучший уровень эффективности и масштабируемости.

2.1. Топология коммуникационной сети

Конечно-разностные схемы на регулярных сетках, в число которых входит FDTD, используют модель параллелизации с распределённой памятью на основе декомпозиции объёма модели и последующего обмена данными вдоль границ разбиения с помощью MPI. Весь объём модели разбивается на прямоугольные области (2D или 3D), каждая область обрабатывается отдельным узлом суперкомпьютера (который, в рамках настоящей работы, может быть и отдельным процессорным ядром, и многопроцессорной системой с ускорителями). Для выполнения вычислений вдоль границ выделенного ему объёма каждому узлу необходимо обмениваться данными с узлами, обрабатывающими смежные объёмы. Требование хорошего масштабирования программы для такого алгоритма накладывает ограничения на топологию коммуникационной сети, объединяющей узлы, а именно: в зависимости от размерности решаемой задачи (2D или 3D) сетевое оборудование должно допускать возможность работы программы в топологии 2D или 3D тора. В этом случае загрузка канала, соединяющего смежные узлы, почти не будет зависеть от общего числа узлов.

Постепенный отказ от древовидных топологий в суперкомпьютерном моделировании физических явлений вызван сложностями при поэтапном наращивании мощности подобных систем. Кроме того, топологии 5D тор и 6D гиперкуб обладают минимальной стоимостью за единицу пропускной способности [27] (почти на порядок меньше, чем у древовидной топологии) и характеризуются большей устойчивостью к отказам узлов сети. Самый мощный на ноябрь 2011 года [24] 10-ти петафлопсный суперкомпьютер «K computer» использует топологию 6D сетка/тор [28], которую можно представить в виде 12-ти 3D торов, поэлементно связанных в частично вырожденный гиперкуб. Самый мощный на июнь 2012 года 20-ти петафлопсный суперкомпьютер «Sequoia» использует топологию 5D тор. Таким образом, можно сделать вывод о том, что суперкомпьютер экзафлопсной мощности, скорее всего, будет предоставлять необходимую для рассматриваемого алгоритма топологию 2D и 3D тора в виде подмножества своей топологии.

2.2. Балансировка нагрузки

При выполнении программ в суперкомпьютерном окружении важным является вопрос балансировки нагрузки или, другими словами, необходимость обеспечить равномерную загрузку вычислениями узлов суперкомпьютера. Для метода FDTD это особенно актуально, так как каждый узел может приступить к выполнению нового шага итерации (к вычислению электромагнитных полей в следующий момент времени) только после того, как он и его ближайшие соседи завершили расчёт для текущего шага.

Можно указать на две основные внешние причины разбалансировки вычислений:

(1) Разная вычислительная мощность узлов суперкомпьютера.
(2) Конкуренция за ресурсы узла среди программ нескольких пользователей, включая конкуренцию за ресурсы с системными процессами.

Разная вычислительная мощность узлов, как правило, обусловлена либо специализацией узлов (под вычисления, требующие, например, больших объёмов оперативной памяти, большой пропускной способности файловой системы, наличия специализированных ускорителей вычислений; под графический вывод результатов; под отладку параллельных программ и т.д.), либо возникает как результат увеличения мощности суперкомпьютера (после добавления новых узлов, содержащих более быстрые процессоры, больше памяти и т.д.).

В целом можно предположить, что с внешними причинами разбалансировки довольно успешно должны бороться менеджеры ресурсов — программы, отвечающие за выделение ресурсов суперкомпьютера для запуска конкретной задачи пользователя. Однотипные узлы суперкомпьютера группируются в разделы, и пользователь может запросить у менеджера ресурсов для запуска своей задачи узлы внутри такого раздела, тем самым гарантировав запуск своей программы в среде, где узлы номинально имеют одинаковую производительность. Кроме того, пользователь может запрашивать у менеджера ресурсов узлы для эксклюзивного использования, тем самым в корне исключив конкуренцию за ресурсы узла с другими пользователями.

Предполагая, что системные процессы в среднем одинаково снижают эффективную производительность для одинаковых узлов, следует учитывать возникающие по этой причине возможные кратковременные изменения эффективной производительности узла. Такие изменения должны быть учтены и, по возможности, скомпенсированы на этапе разработки алгоритма, например, за счёт использования максимально локальных блокировок для синхронизации данных между узлами.

2.3. Пуск и завершение моделирования, вывод результатов

Отдельным вопросом, касающимся не только моделирования интегральных схем нанофотоники, является пуск и завершение моделирования на суперкомпьютерах. По мере увеличения числа узлов, запрашиваемых у менеджера ресурсов, увеличивается и время запуска программы. Так как запуск моделирования на большом числе узлов суперкомпьютера может занимать несколько десятков минут, то в дальнейшем будет предполагаться, что время, затрачиваемое далее программой моделирования, в несколько раз или более превышает время запуска (в противном случае, с точки зрения общего времени счёта, выгоднее, в случае возможности, запускать программу на меньшем числе узлов). Такое ограничение на масштабируемость программы явным образом не будет фигурировать при теоретическом анализе масштабируемости алгоритма FDTD, однако его стоит учитывать при практическом использовании.

Завершение моделирования (в целом или какой-то его части), как правило, сопровождается выводом результата в устройство долговременной памяти, для чего традиционно используются накопители на основе магнитных жёстких дисков (HDD). Главным недостатком таких накопителей является низкая пропускная способность и большое время случайного доступа к данным. Использование параллельных файловых систем позволяет избежать усугубления этих недостатков при масштабировании в рамках суперкомпьютера. Появление твердотельных накопителей, напрямую подключаемых к шине PCIe (у существующих экземпляров пропускная способность на порядок больше, а время случайного доступа на два порядка меньше, чем у HDD), позволяет надеяться, что в будущем вывод результатов на суперкомпьютерах станет значительно быстрее. Тем не менее, предполагая, что вывод в устройство долговременной памяти всё равно будет заметно медленнее, чем остальные процессы, при разработке программы необходимо минимизировать выводимый объём данных.

Специфика моделирования методом FDTD при выводе результатов заключается в возможности разделить их на две группы. В первую входит вывод значений моделируемой величины в каждой точке объёма модели на каком-то шаге итерации напрямую в файл. Последовательно выводя, например, таким способом значение компоненты поля через равное число шагов итерации, можно составить анимированое изображение распространения электромагнитной волны. Такой вывод результатов обладает определённой зрелищностью

и наглядностью, однако может существенным образом замедлять процесс моделирования, что и ограничивает возможности по его использованию.

Во вторую группу входит вывод результатов, полученных после обработки данных моделирования. Рассмотрим для примера фильтр в виде группы связанных волноводов (Рис. 1 б, в). Тогда интерес будут представлять спектр электромагнитной волны на входе в фильтр, спектры волн на разных выходах, модовый состав волноводов, добротность этих мод и т.д. Вывод таких результатов требует значительно меньше ресурсов, несёт информацию о характеристиках предполагаемого дизайна готового устройства и часто осуществляется в самом конце моделирования конкретной системы. Кроме того, если результат такого вывода — число (например, средняя ширина пиков в спектре пропускания), то оно может стать критерием автоматической оптимизации параметров модели. В этом случае время на запуск программы в суперкомпьютерном окружении тратится один раз, а число конфигураций модели, для которых выполняется моделирование, может быть большим.

3. Оценка трудоёмкости моделирования

Для того чтобы при моделировании интегральных схем нанофотоники (при изготовлении которых используется широко распространённая планарная технология [29]) выявить основные закономерности распространения в них света, часто оказывается достаточным использование 2D моделей. Количественно точное описание характеристик отдельных элементов в таких схемах может потребовать 3D моделей. Так как схема оценки трудоёмкости и производительности моделирования для 2D и 3D моделей не отличается, то далее значения, специфичные для 3D модели, будут приводиться в фигурных скобках { } сразу после значений для 2D модели.

Для метода FDTD всё пространство модели дискретизируется прямоугольной сеткой, $\Delta x = \Delta y\{= \Delta z\}$. Согласно оригинальному алгоритму Йи [4], каждому узлу такой сетки соответствует $2\{3\}$ значения компоненты вектора напряжённости электрического поля и $1\{3\}$ значения компонент для вектора напряжённости магнитного поля. Кроме того, в самом простом изотропном, немагнитном, линейном, бездисперсионном и т.д. случае каждому узлу сопоставляется только значение диэлектрической проницаемости $\varepsilon(x, y\{, z\})$. Поэтому в памяти необходимо хранить $4\{7\}$ значений для одного узла сетки. Если есть необходимость учитывать большое число физических эффектов, то это число может вырасти на порядок, поэтому в качестве грубой оценки сверху примем 100 значений на узел сетки дискретизации (как для 2D, так и для 3D модели, различие между ними здесь — не принципиально).

Общее число узлов сетки $N_{\text{total}} = N_x \times N_y\{\times N_z\}$ можно получить из фактических размеров объёма модели и плотности узлов (числа узлов на единицу длины), достаточной для устойчивости модели. При этом все размеры в системе удобно выражать через длину изучаемой электромагнитной волны λ, а систему единиц выбрать так, чтобы скорость света была равна единице.

Максимальный шаг сетки дискретизации определяется из условия устойчивости явного численного решения по критерию Куранта—Фридрихса—Леви [30]. На практике для метода FDTD это означает, что для простых физических ситуаций в методе FDTD шаг дискретизации должен быть меньше, чем $\lambda/20$; в случае наличия в системе анизотропии и дисперсии — менее $\lambda/40$; при необходимости учёта нелинейностей — менее $\lambda/80$. В выбранной системе единиц шаг дискретизации по времени Δt равен шагу дискретизации пространства, т.е. $\Delta t = \Delta x$. Тогда, грубо оценивая достаточную дискретизацию для устойчивости модели в общем случае, будем считать, что плотность сетки составляет 100

шагов на длину волны в пространстве, а плотность дискретизации по времени составляет, соответственно, 100 шагов на период колебаний этой же волны.

Один из наиболее простых элементов интегральной нанофотоники — кольцевой микрорезонатор и фильтр на его основе (Рис. 1 б, в). Для длины волны $\lambda = 1550$ нм при использовании стандартной кремниевой технологии была продемонстрирована [29] работоспособность в составе электронно-оптической интегральной схемы четырёхканального фильтра второго порядка. Площадь фильтра составила приблизительно 600 на 50 мкм, что для используемой длины волны (в кремнии она равна приблизительно 440 нм) эквивалентно размеру 1360 на 114 длин волн. Для однослойной схемы в случае 3D моделирования можно взять глубину моделируемого объёма порядка 10 длин волн (сюда входят как окружающие элемент слои, так и объём, необходимый для граничного поглощающего условия).

В рассматриваемом [29] фильтре радиус колец для каждого частотного канала последовательно менялся на 10 нм $\approx \lambda/50$, минимальные зазоры между волноводами равнялись 80 нм. Чтобы при моделировании учесть изменения геометрических размеров $\lambda/50$, плотность сетки дискретизации должна быть приблизительно на порядок больше. Поэтому в дальнейшем будет использоваться значение 1000 шагов на длину волны (или период колебаний для дискретизации по времени).

Итого, для полного расчёта небольшого элемента интегральной схемы нанофотоники необходимо хранить в памяти

$$N_\lambda \times \rho^{2\{3\}} \times N_{\text{point}} = 1,5 \cdot 10^{5\{6\}} \times 10^{6\{9\}} \times 10^2 = 1,5 \cdot 10^{13\{17\}}$$

значений, где N_λ — размер системы в $\lambda^{2\{3\}}$, ρ — линейная плотность дискретизации, N_{point} — число значений на узел. Если значения представлены числом с плавающей запятой двойной точности, то модель потребует около 30 ТБ оперативной памяти для 2D случая и 300 ПБ для 3D случая, что соответствует или превышает по вычислительным требованиям параметры самых производительных компьютеров TOP500 [24] на сегодняшний день. Существенно упрощая физическую модель, применяя изощрённые схемы дискретизации, подбирая размеры модели кратными сетке дискретизации и т.д., можно уменьшить требования по памяти до уровня суперкомпьютеров терафлопсного уровня.

Минимальный объём вычислений для полной модели рассматриваемого фильтра получается из условия прохождения волны через весь объём модели. В данном случае волне после попадания в фильтр надо пройти расстояние около 2000 длин волн, что при выбранной дискретизации по времени даст около $2 \cdot 10^6$ шагов по времени для каждой точки модели (как для 2D, так и для 3D, так как волна в любом случае распространяется в плоскости интегральной схемы). Перемножая общий объём модели на число шагов (и считая, что каждое значение в среднем используется несколько раз на каждом шаге), получаем трудоёмкость около $10^{20\{24\}}$ операций. Кроме того, в полноразмерной схеме элементарных оптических компонент может быть значительно больше одной. Таким образом, для полного моделирования интегральной схемы нанофотоники необходим суперкомпьютер экзафлопсного (а, возможно, и зеттафлопсного) уровня.

4. Оценка масштабируемости алгоритма

Эффективность ускорения алгоритма можно моделировать, используя предположения закона Амдала [25] (для разных, но фиксированных размеров задачи моделировать ускорение в зависимости от числа узлов) или закона Густавсона [26] (моделировать ускорение, когда размер задачи меняется прямопропорционально числу узлов). Практический интерес представляет смешанный случай, так как в периоды высокой загрузки суперкомпьютера рациональным представляется выполнение моделирования с максимальным использованием

ресурсов каждого узла, а в случае, если заметная часть ресурсов суперкомпьютера простаивает, приоритет смещается в сторону минимизации фактического времени выполнения моделирования.

4.1. Устройство узла суперкомпьютера

Эффективная производительность отдельного узла в составе суперкомпьютера определяется в результате действия целого ряда факторов. В рамках настоящей статьи, как было сказано ранее, узел суперкомпьютера может быть и отдельным процессорным ядром, и многопроцессорной системой с GPU ускорителями. Хотя с практической точки зрения разница между этими двумя случаями огромна, при анализе производительности и масштабируемости алгоритма это оказывается малосущественным.

В достаточно общем случае участие узла суперкомпьютера в решении общей задачи моделирования определяется следующими параметрами:

- Параметр вычислителя: t_{step} — эффективное время, которое узел суперкомпьютера тратит только на то, чтобы выполнить одну итерацию для одного узла сетки дискретизации. Сам по себе этот параметр зависит от большого числа факторов. Например, чем больше число учитываемых моделью физических эффектов (анизотропия, дисперсия и т.д.), тем t_{step} больше. Сложным образом t_{step} может зависеть от общего числа узлов сетки, которые итерирует один узел суперкомпьютера. Наиболее известны изменение t_{step} в результате кэш-эффекта и влияние на итоговое значение t_{step} большой латентности при передаче данных GPU ускорителю (эффективное значение латентности уменьшается с увеличением объёма передаваемых данных). При анализе производительности и масштабируемости алгоритма абсолютное значение t_{step} оказывается малосущественным, важным является его соотношение с остальными параметрами узла.
- Параметр размера оперативной памяти: N_{Mem1Max} — максимальное число узлов сетки дискретизации, все данные для итерирования которых единовременно помещаются в оперативной памяти одного узла суперкомпьютера. Прочие важные характеристики оперативной памяти, как латентность и пропускная способность, косвенно уже учтены в t_{step}.
- Для характеризации коммуникационной сети, объединяющей узлы суперкомпьютера, достаточно (с учётом указанных в разделе 2.1 предположений о топологии сети) двух параметров: t_{startup} — латентность (время инициации соединения), и t_{msg} — «чистое» время на пересылку всех данных, необходимых соседнему узлу сетки дискретизации для новой итерации. Тогда общее время, затрачиваемое узлом суперкомпьютера на пересылку смежному узлу суперкомпьютера всех данных вдоль общей границы, необходимых для новой итерации, определяется как $T_{\mathrm{msg}} = t_{\mathrm{startup}} + t_{\mathrm{msg}} N_{\mathrm{border}}$, где N_{border} — это общее число узлов сетки дискретизации вдоль общей границы.

4.2. Параллельный алгоритм FDTD

При параллельном выполнении алгоритма FDTD после декомпозиции объёма модели отсутствуют последовательные инструкции, поэтому с точки зрения закона Амдала такой алгоритм должен обладать идеальной масштабируемостью. При попытке оценки масштабируемости программы в целом должно быть учтено время, необходимое для запуска программы на суперкомпьютере (см. раздел 2.3), и задержки, связанные с необходимостью обмена данными вдоль границ смежных объёмов. Перед тем как рассматривать вопрос

о влиянии обмена данными на масштабируемость, следует несколько подробнее описать анализируемый параллельный алгоритм FDTD.

Протяжённость границ, вдоль которых необходимо обмениваться данными, сильно зависит от выбора разбиения объёма модели. Оптимальным такое разбиение можно считать тогда, когда протяжённость границ становится минимальной (для выбранного числа P — количества узлов суперкомпьютера, на которых производится моделирование). Дополнительные ограничения на разбиение накладывают требования 1) одинакового объёма для частей разбиения (следует из требований балансировки нагрузки) и 2) прямоугольность частей разбиения и совмещение их углов (т.е. угол части разбиения не может приходиться на какое-либо место границы другой части разбиения, отличное от угла). Второе требование возникает из необходимости последующего сопоставления разбиения объёма модели с топологией коммуникационной сети суперкомпьютера.

Так как в случае суперкомпьютера $P \gg 1$, то примером разбиения, близкого к оптимальному, будет разбиение прямоугольного объёма модели на (почти) квадратные {кубические} области одинакового размера с длиной ребра N_{one} узлов сетки дискретизации. Тогда объём вычислений (в расчёте на один узел суперкомпьютера) составит $(N_{\mathrm{one}})^{2\{3\}}$ узлов сетки, а для выполнения нового шага итерации необходимо будет получить данные по $(N_{\mathrm{one}})^{\{2\}}$ узлу сетки вдоль границы от каждого из четырёх {шести} ближайших соседей.

Для такого обмена данными между смежными узлами суперкомпьютера рекомендуется использование удалённого доступа к памяти (RMA) с активной синхронизацией, описанного в стандарте MPI2. При наличии аппаратной поддержки (предоставляемой большинством производителей оборудования для сетей Infiniband) это позволяет добиться увеличения эффективности масштабирования на величину до 10% [31]. При первичной реализации алгоритма лучше использовать асинхронные сообщения из первой версии стандарта MPI, что упрощает общее восприятие кода программы и обладает более высокой переносимостью между различными реализациями стандарта MPI. Тогда параллельный алгоритм FDTD для одного шага итерации будет следующим (с момента, когда обновление собственных данных по предыдущему шагу завершено):

(1) Подготовка данных для пересылки по всем границам с копированием данных во временный буфер программы.
(2) Запуск асинхронного обмена сообщениями.
(3) Обработка внутренней части вычислительного объёма.
(4) Ожидание окончания обмена сообщениями.
(5) Обработка граничной части вычислительного объёма.
(6) Подготовка к следующему шагу итерации.

Заведомо паразитным и относительно медленным (с возможной низкой эффективностью использования кэш-памяти начиная с некоего размера вычислительного объёма) является первый шаг данного алгоритма, приводящий к двойному копированию (или тройному, в зависимости от реализации стандарта MPI). Копирование во временный буфер необходимо, так как массивы данных, пересылаемые по MPI, должны быть расположены в адресном пространстве оперативной памяти непрерывным образом, что для 2D{3D} случая верно (в лучшем случае) только для двух границ из 4{6}. Второе копирование производится из этого временного буфера программы в буфер отправки сообщения внутри реализации MPI, либо, если последний занят, в промежуточный буфер MPI. Третье копирование может потребоваться для переноса данных из промежуточного буфера MPI в буфер отправки сообщения. Обсуждение эффективности и производительности подобной схемы выходит за рамки настоящей статьи и остаётся на совести разработчиков конкретной реализации MPI.

4.3. Анализ масштабируемости и эффективности алгоритма

Предложенный алгоритм не использует никаких общих для всех узлов суперкомпьютера ресурсов, глобальных блокировок для синхронизации, не имеет общего управляющего процесса. Алгоритм сам по себе никак не зависит от общего числа узлов суперкомпьютера и, следовательно, должен обладать масштабируемостью, близкой к идеальной. Единственным фактором, ограничивающим масштабируемость программы моделирования, является необходимость предварительной декомпозиции модели между узлами суперкомпьютера, которая состоит из оптимального разбиения объёма модели на части и последующего их сопоставления с узлами суперкомпьютера.

Оценивая время, необходимое для такой декомпозиции, можно отметить, что даже для относительно сложных вариантов декомпозиции оно, скорее всего, будет значительно меньше времени, необходимого менеджеру ресурсов суперкомпьютера для запуска программы (см. раздел 2.3). Это связано с тем, что каждый узел может выполнять разбиение по одной и той же подпрограмме (зная общее число узлов и своё положение в топологии коммуникационной сети) локально, без использования сравнительно медленного взаимодействия с другими узлами суперкомпьютера.

Основная идея, заложенная в предлагаемом алгоритме, заключается в сокрытии расходов на коммуникацию и синхронизацию данных за счёт времени, необходимого для обновления значений во внутренней области части объёма декомпозиции. Обмен данными вдоль границ является асинхронным, так что сами по себе операции запуска обмена границами на общее время моделирования влиять почти не должны. Поэтому основным источником снижения эффективности отдельного узла суперкомпьютера может оказаться ожидание окончания обмена сообщениями. Однако в случае, если время, необходимое для обновления значений во внутренней части объёма $T_{\text{internal}} \approx t_{\text{step}}(N_{\text{one}})^{2\{3\}}$, заметно больше, чем время, необходимое для обмена сообщениями $T_{\text{external}} = t_{\text{startup}} + t_{\text{msg}}(N_{\text{one}})^{\{2\}}$, то фазы ожидания (и сопутствующего «простоя») практически отсутствуют, и вычислительные ресурсы узла будут использоваться с эффективностью, близкой к максимальной.

В абстрактном случае для любых фиксированных t_{step}, t_{startup} и t_{msg} всегда найдётся такое N_{one}, чтобы выполнялось **условие максимальной эффективности** работы алгоритма:

$$\frac{T_{\text{internal}}}{T_{\text{external}}} > 1$$

С учётом вышеизложенного несложно сформулировать критерий, который позволит определить, будут ли максимально эффективно использоваться вычислительные возможности узла и, следовательно, будет ли возможна идеальная масштабируемость предлагаемого алгоритма. А именно, если в результате разбиения объёма модели размеры частей декомпозиции таковы, что для каждого узла выполняется условие максимальной эффективности, то в результате стоит ожидать идеальной масштабируемости алгоритма.

Рассмотрим зависимость эффективности алгоритма от числа участвующих в моделировании узлов суперкомпьютера P_{total} для фиксированного размера задачи (случай закона Амдала [25]). Тогда общее число узлов сетки дискретизации, приходящихся на один узел суперкомпьютера, будет равно $N_{\text{oneTotal}} \approx N_{\text{total}}/P_{\text{total}}$. Время обновления значений внутренней части объёма моделирования $T_{\text{internal}} \approx t_{\text{step}}N_{\text{oneTotal}} = t_{\text{step}}N_{\text{total}}/P_{\text{total}}$, т.е. оно будет убывать обратно пропорционально числу задействованных для вычислений узлов суперкомпьютера. Время, необходимое для обмена сообщениями, будет убывать медленнее, как $T_{\text{external}} = t_{\text{startup}} + t_{\text{msg}}\sqrt[2\{3\}]{N_{\text{oneTotal}}} = t_{\text{startup}} + t_{\text{msg}}\sqrt[2\{3\}]{N_{\text{total}}/P_{\text{total}}}$ и для больших P_{total} асимптотически стремиться к t_{startup}. Другими словами, для любых фиксированных t_{startup}, t_{msg}, t_{step}, N_{total} всегда найдётся такое $P_{\max}$, что:

- Если число узлов суперкомпьютера, участвующих в моделировании, меньше этого числа, то от предлагаемого алгоритма можно ожидать (при достаточно большом N_{one}) идеальной масштабируемости и максимальной эффективности использования вычислительных возможностей каждого узла.
- Если число узлов суперкомпьютера больше этого числа, то эффективность может оказаться заметно хуже максимальной. Дальнейшее увеличение числа узлов приведёт к ещё большему падению эффективности и, следовательно, большему отклонению от идеальной масштабируемости. Говоря об общем времени, затраченном на выполнение алгоритма, можно отметить, что с увеличением числа узлов P стоит ожидать его монотонного уменьшения до некоторого фиксированного значения, что, по большому счёту, обусловлено латентностью сети. Другими словами, в предельном случае очень больших P на каждом шаге основное время узел суперкомпьютера будет тратить на инициацию соединения со смежными узлами.

Существует минимальное число узлов P_{min}, необходимое для запуска моделирования. Из ограничений по оперативной памяти, доступной одному узлу суперкомпьютера, следует, что $P_{\mathrm{min}} > N_{\mathrm{total}}/N_{\mathrm{Mem1Max}}$.

В рамках предположений закона Густавсона [26] при увеличении числа узлов $P_{\mathrm{total}} > 1$ всегда будет наблюдаться линейная масштабируемость. Можно ожидать, что эффективность, напрямую связанная со значением отношения $T_{\mathrm{internal}}/T_{\mathrm{external}}$, при изменении P_{total} останется постоянной.

Следует ещё раз подчеркнуть, что вышесказанное справедливо только для случая, когда параметры каждого узла и связывающей его с окружением системной сети не меняются с изменением числа P_{total}. Если, например, не выполняются изложенные в разделе 2.1 предположения о топологии, то с ростом числа узлов может начать расти латентность сети и падать её эффективная пропускная способность, а это сделает недостижимой идеальную масштабируемость даже для случая очень больших N_{one}.

4.4. Влияние внешних факторов

Реальная ситуация с масштабируемостью алгоритма должна быть несколько хуже, что может быть обусловлено, в первую очередь, стохастическими изменениями эффективной производительности узлов суперкомпьютера (например, в результате работы системных процессов). Падение производительности при этом, скорее всего, будет незначительным и измеряться десятками (или даже единицами) процентов. Устойчивость алгоритма к локальным изменениям эффективной производительности узлов предположительно будет осуществляется в рамках двух механизмов компенсации:

(1) Малые изменения имеют тот же эффект, что и небольшие изменения в латентности коммуникационной сети, и их действие компенсируется по тому же механизму, по которому происходит сокрытие расходов на коммуникации. Вследствие стохастической природы таких изменений их усреднённое влияние на вычисления для каждого узла суперкомпьютера будет приблизительно одинаковым и не приведёт к рассинхронизации вычислений между узлами.

(2) Если падение эффективной производительности достаточно велико и продолжительно, то в такой ситуации окружающие узлы могут начать проводить заметное время в ожидании завершения коммуникаций с этим узлом. Однако влияние на общее замедление из-за такого сбоя будет убывать с увеличением расстояния до него. Время разового простоя каждого следующего, более удалённого узла будет на $T_{\mathrm{internal}} - T_{\mathrm{external}}$ меньше, чем у узла, расположенного ближе к месту сбоя. В итоге в пространстве, где по одной из осей отложено локальное время в каждом узле,

возникнет воронка с центром в узле, в котором произошёл сбой, а глубина воронки соответствует времени, на которое этот узел отстал от общего времени. Следующий аналогичный сбой, расположенный в другом месте, приведёт к отставанию только узлов, расположенных в непосредственной близости от него. В случае, если объём модели достаточно велик, то такие сбои с высокой вероятностью не будут перекрываться. Таким образом, для большого числа узлов влияние таких сбоев окажется заметно слабее прямой пропорциональности их числу и силе. В частности, если все сбои оказались достаточно разнесены между собой, то общее время счёта увеличится на время одного сбоя — самого сильного.

В случае, если один из узлов суперкомпьютера, на котором выполнялось моделирование, полностью выходит из строя, то происходит аварийное завершение работы программы. Введение избыточности в алгоритм для повышения устойчивости к отказу узлов суперкомпьютера в общем случае неизбежно снизит его эффективность. Однако возможна ситуация, когда большая часть данных для расчёта может быть достаточно быстро восстановлена из исходных условий задачи моделирования. Тогда, при наличии достаточного количества оперативной памяти, возможно резервное копирование прочих данных на смежный узел. В случае высокой трудоёмкости вычислений и быстрой коммуникационной сети может оказаться, что условие максимальной эффективности по-прежнему выполняется, а значит такое резервирование не влияет на эффективность алгоритма и его масштабируемость. В этой ситуации аварийное завершение работы может включать в себя, например, сохранение в устройство долговременной памяти данных по всей модели, необходимых для возобновления моделирования. После устранения неполадки отказавшего узла суперкомпьютера (или, например, после запуска программы в исправной части суперкомпьютера) моделирование возобновляется без потерь результатов, полученных до аварийного завершения работы.

4.5. Численная оценка перспектив параллелизации

Оценим возможную масштабируемость и эффективность алгоритма для реально существующих систем. Для расчётов будем полагать, что узел суперкомпьютера состоит из одного процессорного ядра, пропорциональной доли оперативной памяти и нужного числа сетевых портов. Современные потребительские (на начало 2012 года) процессоры обеспечивают производительность около 20 гигафлопс на ядро (из чего получается $t_{\mathrm{step}} \approx 0,05\ \frac{\text{нс}}{\text{операция}} \times 100\ \frac{\text{значений}}{\text{узел}} \times 10 \frac{\text{операций}}{\text{значение}} = 50\ \frac{\text{нс}}{\text{узел}}$), материнские платы поддерживают до 64 Гб оперативной памяти (около 10 Гб в пересчёте на одно ядро, т.е. $N_{\mathrm{Mem1Max}} \approx 5 \cdot 10^7$ узлов двойной точности). Рассмотрим ситуации для двух вариантов сети: потребительской 1000BASE-T Gigabit Ethernet (GbE) и серверной 4x DDR Infiniband (IB). Ориентировочное значение t_{startup} составляет 100 мкс для GbE и 1 мкс для IB. Для GbE установившаяся скорость соединения составляет 1 Гбит/с $= 125$ Мб/с $= 65 \cdot 10^6$ значений/с $= 65 \cdot 10^4$ узлов/с, а значит $t_{\mathrm{msg(GbE)}} = 1,5$ мкс/узел. Для IB скорость в 16 раз выше, поэтому ориентировочно $t_{\mathrm{msg(IB)}} = 0,1$ мкс/узел.

Критическое значение N_{one}, при котором начинает выполняться условие максимальной эффективности алгоритма, получается из решения уравнения $T_{\mathrm{internal}} = T_{\mathrm{external}}$. После подстановки получаем уравнение для 2D случая $t_{\mathrm{step}} N_{\mathrm{one}}^2 - t_{\mathrm{msg}} N_{\mathrm{one}} - t_{\mathrm{startup}} = 0$, положительный корень которого и является искомым значением. Аналогичным образом получается уравнение для случая 3D, где смысл имеет только вещественный корень. Результаты расчёта в таблице 1 свидетельствуют о том, что в случае идеальной реализации предлагаемого алгоритма условие максимальной эффективности будет удовлетворено с хорошим запасом

	2D	3D
GbE	62	6
IB	32	4

ТАБЛИЦА 1. Критическое значение N_{one} для условия максимальной эффективности предлагаемого параллельного алгоритма FDTD

(надо сравнивать квадрат{куб} полученного N_{one} с N_{Mem1Max}). Для менее идеализированного случая реальные величины скорее всего будут в несколько раз больше, но даже после увеличения их на порядок в силе остаётся следующий вывод: возможно создание эффективной и масштабируемой программы на уже существующей технической базе. Более того, возможно построение высокопроизводительного вычислительного кластера для моделирования интегральных схем нанофотоники, использующего относительно недорогие комплектующие потребительского сегмента компьютерного рынка.

5. Заключение

Оценка трудоёмкости моделирования интегральных схем нанофотоники методом FDTD показала необходимость реализации параллельного алгоритма, обладающего высокими эффективностью и уровнем масштабирования, что в перспективе позволит использовать для такого моделирования суперкомпьютеры экзафлопсного уровня производительности. Приводится пример подобного алгоритма, дана оценка его возможной масштабируемости. Обосновываются требования к балансу характеристик узла суперкомпьютера для достижения экзафлопсной производительности при условии максимальной эффективности алгоритма. Показано, что параметры современной технической базы для суперкомпьютерных вычислений соответствуют этим требованиям.

На основе изложенного подхода в дальнейшем планируется практическая реализация параллельного FDTD. Первые же результаты на этом пути показали, что для ряда задач можно достичь эффективности параллельной программы более 100%. Например, для конкретной 2D модели время счёта на 16 узлах кластера оказалось в 24 раза меньше, чем на одном узле (подобное поведение связано с кэш эффектом). В целом это свидетельствует о перспективности данного направления разработки.

Авторы выражают благодарность Министерству образования РФ за финансовую поддержку исследований в рамках гранта Правительства Российской Федерации для государственной поддержки научных исследований, проводимых под руководством ведущих учёных в российских образовательных учреждениях высшего профессионального образования, а также Бухановскому А.В. за обсуждение работы.

Литература

[1] Taflove A., Hagness S. C. Computational Electrodynamics: the Finite-Difference Time-Domain Method. — 3rd edition. — Artech House, 2005.

[2] Гельфонд А. О. Исчисление конечных разностей. — 2-ое изд. — М. : ФМЛ, 1959.

[3] Самарский А. А. Введение в теорию разностных схем. — М. : Наука, 1971.

[4] Yee K. Numerical Solution of Initial Boundary Value Problems Involving Maxwell's Equations in Isotropic Media // IEEE Trans. on Antennas and Propagation.— 1966.— V. AP14, №3. — P. 302–307.

[5] Mur G. Absorbing Boundary Conditions for the FiniteDifference Approximation of the TimeDomain ElectromagneticField Equations // IEEE Trans. on Electromagnetic Compatibility. — 1981. — V. EMC23, №4. — P. 377–382.

[6] Higdon R. L. Absorbing Boundary Conditions for Difference Approximations to the MultiDimensional Wave Equation // Mathematics of Computation. — 1986. — V. 47, № 176. — P. 437–459.

[7] Ramahi O. M. The Concurrent Complementary Operators Method for FDTD Mesh Truncation // IEEE Trans. on Antennas and Propagation. — 1998. — V. 46, № 10. — P. 1475–1482.
[8] Berenger J.P. A Perfectly Matched Layer for the Absorption of Electromagnetic Waves // J. of Computational Physics. — 1994. — V. 114, № 2. — P. 185–200.
[9] Gedney S. D. An Anisotropic PML Absorbing Media for the FDTD Simulation of Fields in Lossy and Dispersive Media // Electromagnetics. — 1996. — V. 16, № 4. — P. 399–415.
[10] Roden J. A., Gedney S. D. Convolution PML (CPML): An efficient FDTD implementation of the CFS-PML for arbitrary media // Microwave and Optical Technology Lett. — 2000. — V. 27, № 5. — P. 334–339.
[11] Teixeira F. L. On aspects of the physical realizability of perfectly matched absorbers for electromagnetic waves // Radio Science. — 2003. — V. 38, № 2. - P. 8014.
[12] Moore G. Cramming more components onto integrated circuits // Electronics Magazine. — 1965. — V. 38, № 8. — P. 4.
[13] Yu W., Mittra R., Su T. et al. Parallel FiniteDifference TimeDomain Method. — Artech House, 2006.
[14] Inan U. S., Marshall R. A. Numerical Electromagnetics The FDTD Method. — Cambridge University Press, 2011.
[15] URL: http://www.cvel.clemson.edu/modeling/index.html.
[16] Bondeson A., Rylander T., Ingelström P. Computational Electromagnetics. — Springer, 2005.
[17] Yu W., Yang X., Liu Y. et al. Advanced FDTD Methods: Parallelization, Acceleration, and Engineering Applications. — Artech House, 2011.
[18] Ильин В. П. Об экзапроблемах математического моделирования // Тр. Параллельные вычислительные технологии — Уфа, Россия, 2010.
[19] URL: http://abinitio.mit.edu/wiki/index.php/Meep.
[20] URL: http://www.acceleware.com/fdtdsolvers.
[21] URL: http://www.2comu.com/products_pc_cluster.html.
[22] URL: http://www.photond.com/products/fdtd/fdtd02.htm.
[23] URL: http://www.remcom.com/xf7mpi/.
[24] URL: http://www.top500.org.
[25] Amdahl G. M. Validity of the SingleProcessor Approach to Achieveing Large Scale Computing Capabilities // In Proc. AFIPS Conference. — 1967.
[26] Gustafson J. L. Reevaluating Amdahl's Law // Communications of the ACM. — 1988. — V. 31. — P. 532–533.
[27] Hospodor A., Miller E. L. Interconnection Architectures for PetabyteScale HighPerformance Storage Systems // 12th NASA Goddard Conference on Mass Storage Systems and Technologies (MSST2004). — 2004.
[28] Ajima Y., Sumimoto S., Shimizu T. Tofu: A 6d Mesh/torus Interconnect for Exascale Computers // Computer, IEEE Computer Society. — 2009. — V. 42, № 11. — P. 36–40.
[29] Orcutt J. S., Khilo A., Holzwarth Ch. W. et al. Nanophotonic integration in stateoftheart CMOS foundries // Optics Express. — 2011. — V. 19, № 3. — P. 2335–2346.
[30] Курант Р., Фридрихс К., Леви Г. О разностных уравнениях математической физики // УМН. — 1941. — № 8. — С. 125–160.
[31] Potluri S., Lai P., Tomko K. et al. Quantifying Performance Benefits of Overlap using MPI2 in a Seismic Modeling Application // Proc. of the 24th ACM International Conference on Supercomputing. — 2010.