МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М. В. ЛОМОНОСОВА

ФИЗИЧЕСКИЙ ФАКУЛЬТЕТ
КАФЕДРА ОБЩЕЙ ФИЗИКИ И ВОЛНОВЫХ ПРОЦЕССОВ
МЕЖДУНАРОДНЫЙ УЧЕБНО-НАУЧНЫЙ ЛАЗЕРНЫЙ ЦЕНТР
НОЦ «СУПЕРКОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ»

Отчетная работа

Обзор и сравнительный анализ параллельных численных алгоритмов решения нелинейного уравнения квазиоптики

Дергачёв А. А.
Ефимов О. В.
Сметанина Е. О.

Москва, 2009

Содержание

1. Введение

Распространение мощных фемтосекундных лазерных импульсов в атмосфере сопровождается явлением филаментации, когда существенная часть энергии излучения концентрируется в «горячих точках» в поперечном сечении импульса, которые наблюдаются в виде тонких протяженных нитей. Впервые светящаяся нить, или филамент, была зарегистрирована в 1965 году при фокусировке наносекундных лазерных импульсов в кювету с органическими жидкостями [1]. С созданием фемтосекундных лазерных установок стало возможным получение протяженных (до нескольких сотен метров) филаментов при распространении излучения в газовых средах, в частности, в атмосфере. Краткий обзор этих и других экспериментальных результатов дан в [2].

Причиной начала формирования нитей-филаментов является эффект Керра, вызывающий самофокусировку пучка в среде. Самофокусировка имеет место, если мощность пучка превосходит некоторую критическую P_{cr}, зависящую как от параметров среды (линейного и нелинейного коэффициентов преломления), так и от параметров пучка (таких как длина волны излучения и радиус, а также от формы пучка, степени его осесимметричности и так далее). С простейшей теорией самофокусировки можно ознакомиться в [3]. Рост интенсивности останавливается в нелинейном фокусе за счет дефокусирующего действия наведенной излучением плазмы, возникающей в результате действия нескольких механизмов, прежде всего многофотонной и туннельной ионизации. Обычно филамент формируется на оси пучка на расстоянии, соответствующем положению нелинейного фокуса для центрального, наиболее мощного временного слоя. На его положение могут оказывать влияние такие факторы, как флуктуации показателя преломления в турбулентной атмосфере и фазовые флуктуации в начальном пучке.

2. Постановка задачи

В простейшем случае уравнение, описывающее самофокусировку гауссового пучка, записывается в виде:

$$\begin{cases} 2ik\dfrac{\partial E}{\partial z} &= \dfrac{\partial^2 E}{\partial x^2} + \dfrac{\partial^2 E}{\partial y^2} + \dfrac{2k^2}{n_0} n_2 \left|E\left(x,y,z\right)\right|^2 E\left(x,y,z\right) \\ E(x,y,0) &= E_0 \exp\left\{-\dfrac{x^2+y^2}{2a_0^2}\right\}, \quad (x,y) \in [-l,l]^2 \end{cases} \tag{1}$$

Здесь $E\left(x,y,z\right)$ — напряженность электрического поля, k — волновое число, z — координата вдоль оси распространения лазерного пучка, x, y — координаты в поперечном сечении, n_0 и n_2 — линейный и нелинейный коэффициент преломления. В данном уравнении учтены такие физические факторы, как дифракция лазерного пучка и кубическая (по полю) нелинейность. Таким образом, оно описывает начальный этап филаментации, а именно распространение пучка до нелинейного фокуса, когда возросшая интенсивность приведет к плазмообразованию и дефокусировке. Эти процессы требуют введения дополнительных слагаемых в правую часть (1). Однако численное решение нелинейного уравнения квазиоптики представляет и самостоятельный интерес.

После обезразмеривания $E = \tilde{E} \cdot E_0$, $x = \tilde{x} \cdot a_0$, $y = \tilde{y} \cdot a_0$, $z = \tilde{z} \cdot k a_0^2$ система будет выглядеть следующим образом:

$$\begin{cases} 2i\dfrac{\partial \tilde{E}}{\partial \tilde{z}} &= \Delta_\perp \tilde{E} + R\left|\tilde{E}\right|^2 \tilde{E} \\ \tilde{E}(\tilde{x},\tilde{y},0) &= \exp\left\{-\dfrac{\tilde{x}^2+\tilde{y}^2}{2}\right\}, \quad (x,y) \in \left[-\dfrac{l}{a_0}, \dfrac{l}{a_0}\right]^2 \end{cases} \tag{2}$$

Основные проблемы численного моделирования задачи филаментации лазерных импульсов связаны с многомасшабностью задачи. Поперечные масштабы пучка примерно на два порядка превосходят возникающие в нем структуры. В то же время размер расчетной сетки должен на порядок превосходить радиус пучка, чтобы границы сетки не отсекали существенные части пучка, а также, чтобы иметь некоторую «буферную область», в которую могла бы расширяться низкоинтенсивная периферийная часть пучка. В противном случае неизбежно возникновение краевых эффектов, приводящих к искажению решения. Кроме того, на диаметр филамента должно приходиться достаточное количество точек (не менее 10), иначе резкие перепады интенсивности в окрестности филамента будут содержать слишком высокие пространственные частоты, что приведет к невыполнению критерия Найквиста, наложению частот и, как следствие, неадекватности получаемого решения.

Таким образом, количество точек в поперечном сечении достигает 10^4 по каждой поперечной координате. Если учесть теперь, что количество временных слоев составляет величину порядка $10^2 - 10^3$, то общее количество точек достигает величины 10^{11}, а потребность в оперативной памяти — величины порядка 100 Гб.

3. Параллельные алгоритмы решения задачи

Были предложены три метода численного решения поставленной задачи. Один метод основан на использовании явной схемы для уравнения (2). Два других предусмат-

ривают предварительное расщепление по физическим факторам. Интегрирование нелинейного уравнения дифракции сводится к последовательному интегрированию на каждом шаге интегрирования двух уравнений: первое описывает только дифракцию, второе — только нелинейность.

Эти уравнения имеют вид:

$$\begin{cases} 2i\dfrac{\partial E}{\partial z} &= \Delta_\perp E \\ 2i\dfrac{\partial E}{\partial z} &= R\left|E\right|^2 E \end{cases} \tag{3}$$

Интегрирование уравнения для нелинейности не представляет проблем:

$$E(x,y,z_i+\Delta z) = E(x,y,z_i)\exp\left(-\frac{iR}{2}\left|E(x,y,z_i)\right|^2\Delta z\right) \tag{4}$$

Отметим, что поскольку нелинейность считается локальной, то есть набег фазы в точке поперечного сечения зависит только от значения интенсивности поля в этой же точки, поэтому применение метода геометрического параллелизма не вызывает проблем.

Для интегрирование уравнения дифракции была использована неявная численная схема и метод, использующий быстрое преобразование Фурье.

Шаг интегрирования по оси z по мере приближения к нелинейному фокусу должен уменьшаться, поскольку рост интенсивности при самофокусировке приводит к росту нелинейного фазового набега засчет эффекта Керра. Критерием пригодности шага был следующим:

$$\max_{x,y}\frac{R}{2}\left|E(z_i)\right|^2\Delta z \leqslant 0{,}01. \tag{5}$$

3.1. Явная схема

Уравнение (2) переписывается в виде:

$$\Delta E(z_i) = E(z_i+\Delta z) - E(z_i) = \frac{\partial E}{\partial z}\Delta z = \frac{1}{2\mathbf{i}}(\Delta_\perp E(z_i) + R|E(z_i)|^2 E)\Delta z = f(x,y,z)\Delta z \tag{6}$$

Для его решения можно применять как очень нестабильный метод Эйлера, так и заведомо более устойчивые методы из класса «предиктор-корректор», например, метод Рунге-Кутта 4-го порядка. Он состоит в последовательном вычислении значения функции f от некоторых аргументов и последующем усреднении полученного таким образом значения приращения. В случае, если у нас нет поперечных координат, это выглядело бы следующим образом:

$$\begin{aligned} k_1 &= f(E_i, t) \\ k_2 &= f(E_i + \frac{k_1\Delta z}{2}) \\ k_3 &= f(E_i + \frac{k_2\Delta z}{2}) \\ k_4 &= f(E_i + k_3\Delta z) \\ E_{i+1} &= E_i + (k_1 + 2k_2 + 2k_3 + k_4)\Delta z/6 \end{aligned} \tag{7}$$

Соответственно, в нашем случае на каждом шаге нужно рассчитывать значение 4-х матриц $k_i[x_n, y_m]$.

Некоторые модификации данного метода используются другими исследователями этой проблемы. В целом же из-за зависимости функции $f()$ от поперечных координат(через поперечный лапласиан) делает эту схему неустойчивой для решения данной задачи. При этом из критерия Куранта-Фридрихса-Леви следует, что шаг по оси z должен быть меньше чем $c(\Delta x)^2$, что при увеличении количества точек в поперечном сечении приводит к чрезвычайно малому шагу по z. Однако это не ставит крест на алгоритме, так как он может применяться в тех случаях, когда требуется большая точность результатов и шаг заведомо должен быть довольно мал.

3.2. Метод Фурье

Метод Фурье основан на переход к двумерному Фурье-образу матрицы поля:

$$E\left(k_x, k_y, z\right) = \iint E\left(x, y, z\right) e^{-ik_x x - ik_y y}\, dxdy \tag{8}$$

Дифракция в Фурье-представлении будет описываться следующим уравнением:

$$2i\frac{\partial E\left(k_x, k_y, z\right)}{\partial z} = (-k_x^2 - k_y^2) E\left(k_x, k_y, z\right), \tag{9}$$

решение которого дается формулой

$$E\left(k_x, k_y, z + \Delta z\right) = E\left(k_x, k_y, z\right) \exp\left\{\frac{i}{2}(k_x^2 + k_y^2)\left|E\left(k_x, k_y, z\right)\right|^2\right\} \tag{10}$$

Процедура параллельного вычисления двумерного преобразования Фурье бралась из свободно распространяемой библиотеки FFTW [4]. Данная процедура предполагает ленточное распределение матрицы поля по процессам. Кроме того, преобразование Фурье быстрее всего работает на матрицах, размеры которых являются степенями двойки.

Основными этапами выполнения преобразования являются создание плана, который содержит информацию об организации пересылок между процессами, и собственно выполнение преобразования. Функция работает следующим образом. Вначале выполняется Фурье-преобразование по строкам. Этот этап происходит локально на каждом процессе, поскольку процесс содержит в своей оперативной памяти всю строку. Затем происходит транспонирование распределенной матрицы, что связано с обменами данными между всеми процессами (то есть каждый обменивается с каждым). Далее снова выполняется Фурье-преобразование по строкам, которые ранее были столбцами.

Существенной особенностью процедуры является расположение полученных коэффициентов в памяти процессоров. Для преобразования Фурье естественным является транспонированное расположение результата в памяти всех процессов. Этому соответствует ключ FFTW_TRANSPOSED_ORDER функции, реализующей преобразование Фурье. Его альтернативой является ключ FFTW_NORMAL_ORDER, который после выполнения преобразования Фурье проводит дополнительно транспонирование матрицы. Кроме того, один из параметров указанной функции может содержать дополнительный буферный массив для ускорения преобразования. Наконец, при создании плана Фурье-преобразования существует возможность оптимизировать план с целью убыстрения работы функции. Это достигается использованием ключей FFTW_ESTIMATE

(грубая оценка) и FFTW_MEASURE (при этом производятся замеры времени пересылок, выполняемых в Фурье-преобразовании и их оптимизация).

Целью работы были замеры времени работы данного алгоритма в зависимости от размера матрицы, количества процессов, а также от комбинаций указанных ключей.

3.3. Метод с использование неявной схемы

Используемая консервативная разностная схема для исходного уравнения (2) для неравномерных сеток в поперечном сечении была предложена в ([5]). Она выглядит следующим образом:

$$\begin{cases} 2i\dfrac{h_1}{2}\dfrac{\hat{E}_{0,j}-E^k_{0,j}}{\Delta z} = \dfrac{1}{2}\left(\dfrac{\hat{E}_{1,j}-\hat{E}_{0,j}}{h_1}\right)+\dfrac{1}{2}\left(\dfrac{E^k_{1,j}-E^k_{0,j}}{h_0}\right) \\ 2i\dfrac{h_{i+1}+h_i}{2}\dfrac{\hat{E}_{i,j}-E^k_{i,j}}{\Delta z} = \dfrac{1}{2}\left(\dfrac{\hat{E}_{i+1,j}}{h_{i+1}}-\left(\dfrac{1}{h_{i+1}}+\dfrac{1}{h_i}\right)\hat{E}_{i,j}+\dfrac{\hat{E}_{i-1,j}}{h_i}\right) \\ \qquad\qquad +\dfrac{1}{2}\left(\dfrac{E^k_{i+1,j}}{h_{i+1}}-\left(\dfrac{1}{h_{i+1}}+\dfrac{1}{h_i}\right)E^k_{i,j}+\dfrac{E^k_{i-1,j}}{h_i}\right), \quad i=1,\dots N \\ 2i\dfrac{h_N}{2}\dfrac{\hat{E}_{N,j}-E^k_{N,j}}{\Delta z} = \dfrac{1}{2}\left(\dfrac{\hat{E}_{N,j}-\hat{E}_{N-1,j}}{h_N}\right)+\dfrac{1}{2}\left(\dfrac{E^k_{N,j}-E^k_{N,j}}{h_{N-1}}\right) \end{cases} \tag{11}$$

Указанная схема осуществляет расчет «дифракции по x». Аналогичная система разностных уравнений рассчитывает «дифракцию по y». Если шаг сетки равномерный, то есть $h_i = x_i - x_{i-1} = const$, то схема переходит в хорошо известную схему Кранка-Николсона. Учет керровской нелинейности осуществляется в рамках описан в разделе 3.

Указанная система является системой относительно значений поля на промежуточном слое $\hat{E}_{i,j}$. Матрица системы трехдиагональна и решается методом прогонки (ссылка на Калиткина). Рассмотрим вариант параллельной реализации прогонки.

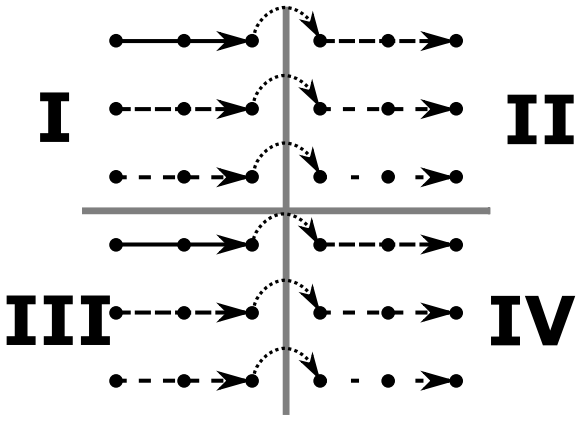

Рис. 1. Метод с использованием неявной схемы. Прямая прогонка.

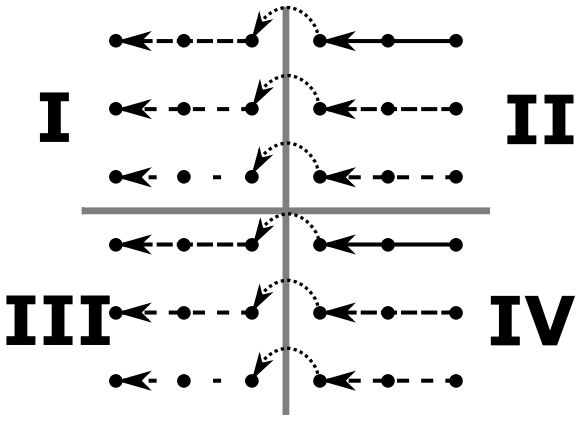

Рис. 2. Метод с использованием неявной схемы. Обратная прогонка.

На рис. 1, 2 черными точками изображены точки расчетной сетки. Вертикальная и горизонтальная светло-серые прямые показывают разбиение матрицы поперечного сечения между процессами. Римские цифры обозначают номера процессов. Цветные

стрелки соответствуют вычислению прогоночных коэффициентов. Серые стрелки отображают обмены данными между процессами.

Первому расчетному шагу соответствуют стрелки красного цвета. Для цикла прямой прогонки их выполняют процессы первого столбца матрицы процессов. Далее эти процессы пересылают посчитанные коэффициенты в граничной точке соседнему справа процессу. Следующий такт отображен стрелками зеленого цвета, далее следуют синие и коричневые стрелки. Таким образом, по каждой строке матрицы процессов запускается конвейерная схема параллельных вычислений.

В цикле обратной прогонки конвейерная схема запускается в обратную сторону (соответствие цвета стрелок и порядка выполнения расчетных операций то же). Пересылке теперь подвергаются расчитанные значения поля.

Отметим две особенности метода. Во-первых, поскольку цикл обратной прогонки запускается после окончания прямой прогонки по всем строкам расчетной сетки, то необходимо хранить в памяти прогоночные коэффициенты для всех строк, то есть необходима дополнительная матрица для прогоночных коэффициентов того же размера, что и матрица поля. Во-вторых, для расчета коэффициентов, фигурирующих в разностной системе (11), необходим обмен значениями поля в граничных областях.

Отметим также, что для высокой эффективности параллельного алгоритма необходимо, чтобы количество строк матрицы поля у отдельного процесса было существенно больше количества процессов в строке матрицы процессов. Действительно, если число строк у отдельного процесса равно N_{loc}, а число процессов в строке матрицы процессов равно q (на рис. 1, 2 $N_{loc} = 3$, $q = 2$), то на цикл прямой прогонки потребуется $N_{loc} + q$ итераций. Таким образом, верхняя оценка на эффективность имеет вид:

$$E_n \leqslant \frac{N_{loc} q^2}{q^2 (N_{loc} + q)} = \frac{1}{1 + q/N_{loc}}. \tag{12}$$

В этом соотношении не учтены потери времени на пересылки и синхронизацию процессов.

4. Результаты

Все реализации были тестированы на корректность на аналитическом решении (свободной дифракции гауссова пучка), и на консервативность.

Представлены результаты замеров времени работы алгоритма с использованием преобразования Фурье при различных комбинациях параметров, выбранных для анализа их влияния на время работы программы. Запуск и замеры времени осуществлялись на кластере СКИФ МГУ «Чебышёв».

Из представленных на рис. 3, 4 графиков видно, что использование более 16 процессоров является неэффективным для размера матрицы $N = 512$, так как время работы программы возрастает по сравнению с временем работы программы при тех же параметрах на 8 процессорах.

Для размера матрицы 2048, как видно из рис. 5, 6 увеличение числа процессоров от 1 до 16 дает ощутимое уменьшение времени работы программы.

При использовании матрицы размером 8192 времена работы программы увеличиваются соответственно, что позволяет увидеть более четко различие во временах работы программы при использовании различных комбинаций флагов.

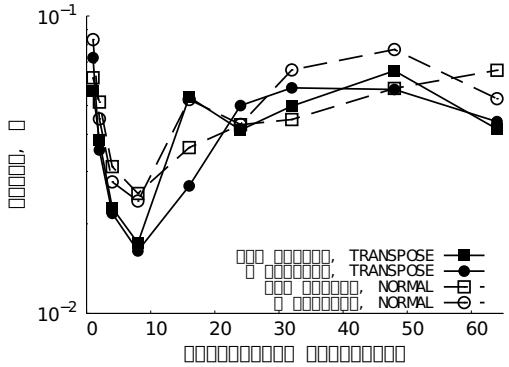

Рис. 3. Время работы Фурье-алгоритма в зависимости от количества процессов. Размер матрицы 512. Флаг FFTW_ESTIMATE.

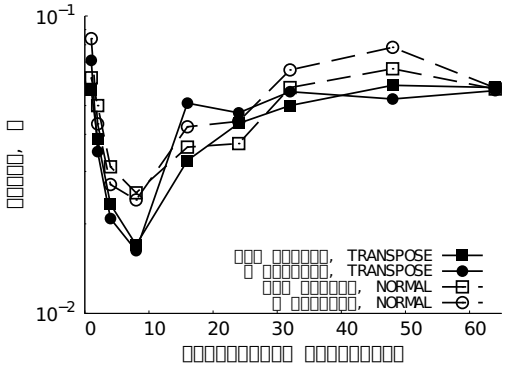

Рис. 4. Время работы Фурье-алгоритма в зависимости от количества процессов. Размер матрицы 512. Флаг FFTW_MEASURE.

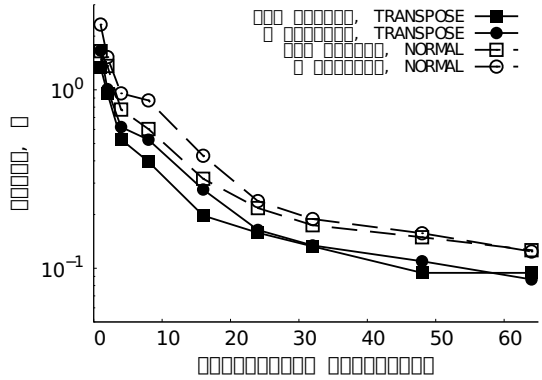

Рис. 5. Время работы Фурье-алгоритма в зависимости от количества процессов. Размер матрицы 2048. Флаг FFTW_ESTIMATE.

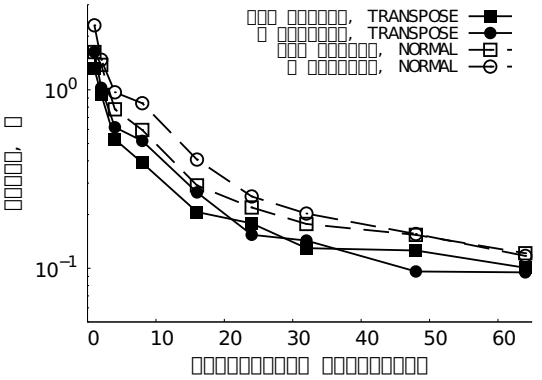

Рис. 6. Время работы Фурье-алгоритма в зависимости от количества процессов. Размер матрицы 2048. Флаг FFTW_MEASURE.

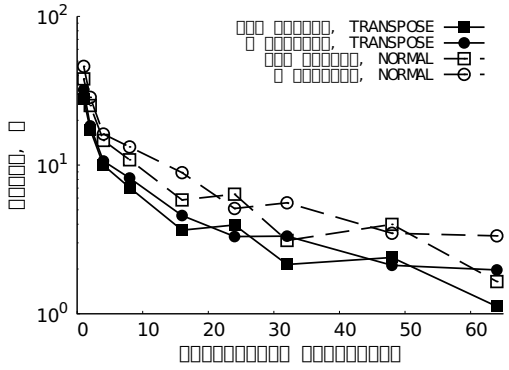

Рис. 7. Время работы Фурье-алгоритма в зависимости от количества процессов. Размер матрицы 8192. Флаг FFTW_ESTIMATE.

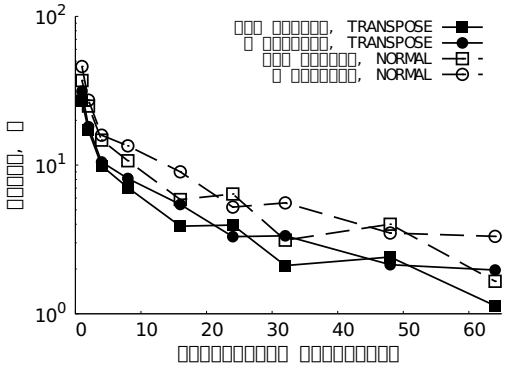

Рис. 8. Время работы Фурье-алгоритма в зависимости от количества процессов. Размер матрицы 8192. Флаг FFTW_MEASURE.

Из графиков на рис. 3–8 видно, что использования флага FFTW_MEASURE не приводит к убыстрению работы алгоритма, что связано прежде всего с малым количеством расчетных узлов. Кроме того, использование буфера не только не привело к ускорению алгоритма, но, наоборот, несколько затормозило его. По представленным результатам был сделан вывод, что лучшее враг хорошего. Скорость работы алгоритма с ключом FFTW_TRANSPOSED_ORDER, как и ожидалось, оказалась выше, чем с ключом FFTW_NORMAL_ORDER.

Было проведено исследование зависимости времени работы алгоритма от использу-

емых ключей компиляции. Результаты представлены в табл. 1.

Опция	$np = 8$	$np = 16$
-O2 (по умолчанию)	9.0 с	5.0 с
-O1	9.2 с	5.3 с
-O3	8.8 с	5.1 с
-Os	9.0 с	5.2 с
-fast	8.9 с	5.2 с

Таблица 1. Зависимость времени такта от опций компиляции.

Таким образом, использование ключей компиляции не дало ощутимого уменьшения времени работы программы.

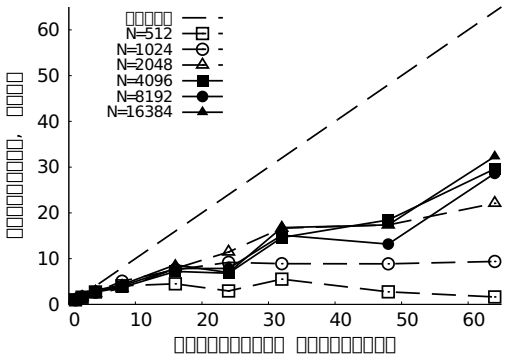

Рис. 9. Ускорение Фурье-алгоритма в зависимости от количества процессов для разных размеров матриц. Сохранение данных в файл не производилось.

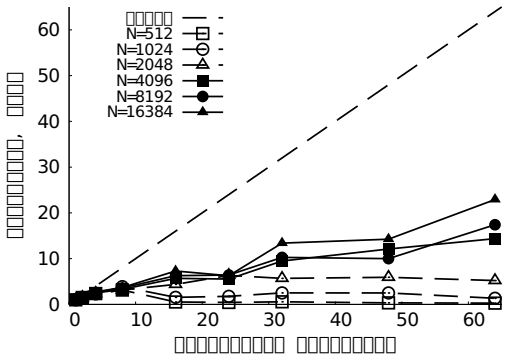

Рис. 10. Ускорение Фурье-алгоритма в зависимости от количества процессов для разных размеров матриц. Сохранение данных в файл производилось на каждом десятом шаге.

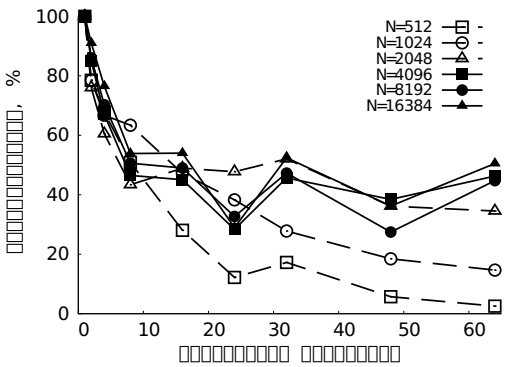

Рис. 11. Эффективность Фурье-алгоритма в зависимости от количества процессов для разных размеров матриц. Сохранение данных в файл не производилось.

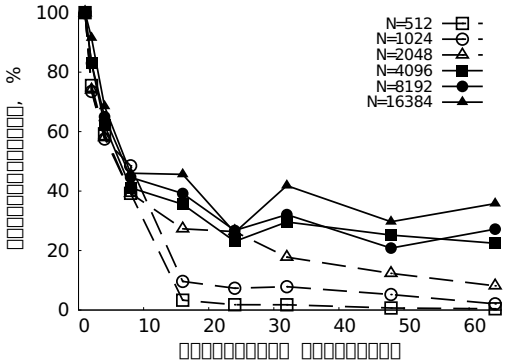

Рис. 12. Эффективность Фурье-алгоритма в зависимости от количества процессов для разных размеров матриц. Сохранение данных в файл производилось на каждом десятом шаге.

Были проведены замеры времени для лучшего набора опций (отсутствие буфера, FFTW_TRANSPOSED_ORDER, FFTW_ESTIMATE). Замеры проводились без сохранения матрицы в файл и с параллельной записью матрицы в файл на каждом десятом шаге. Рассчитанные по полученным данным ускорения и эффективности программ представлены на рис. 9–12. Видно, что для размера матрицы поля, не превосходящего

2048 × 2048, использование более 8 процессов нецелесообразно, поскольку не дает прироста скорости. Эффективность реализации резко падает для размеров матрицы поля 512 × 512 и 1024 × 1024. Для больших матриц эффективность остается значительной: около 50% для случая без сохранения данных и около 30% для случая с сохранением.

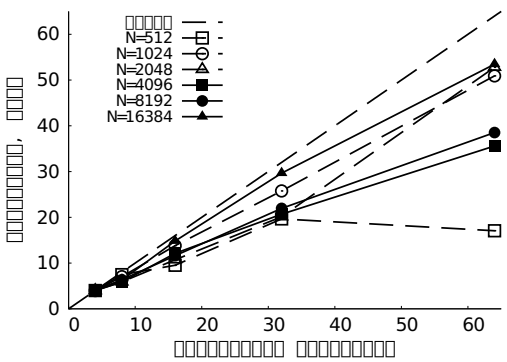

Рис. 13. Ускорение параллельного алгоритма с неявной схемой. СКИФ МГУ. Сохранение данных в файл не производилось.

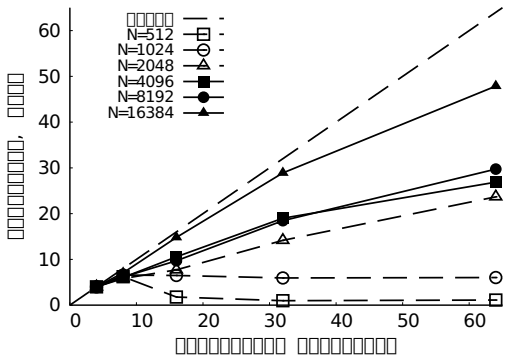

Рис. 14. Ускорение параллельного алгоритма с неявной схемой. СКИФ МГУ. Сохранение данных в файл производилось на каждом десятом шаге.

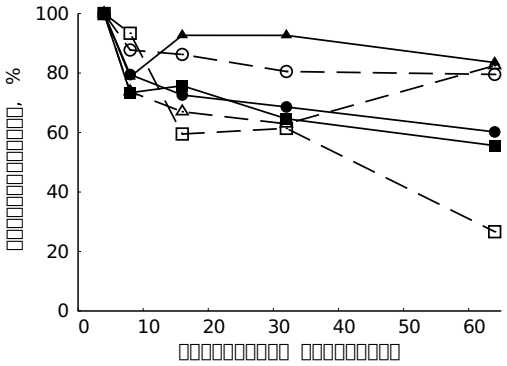

Рис. 15. Эффективность параллельного алгоритма с неявной схемой. СКИФ МГУ. Сохранение данных в файл не производилось.

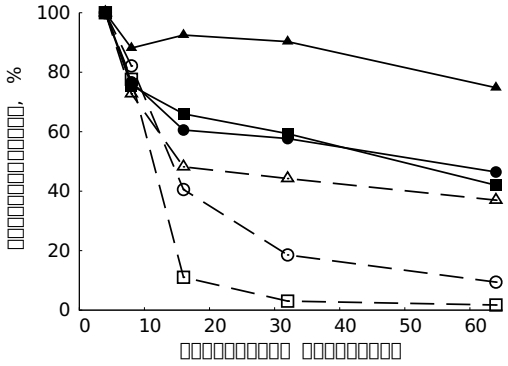

Рис. 16. Эффективность параллельного алгоритма с неявной схемой. СКИФ МГУ. Сохранение данных в файл производилось на каждом десятом шаге.

Для алгоритма, использующего неявную схему, проводились замеры времени работы на кластерах СКИФ МГУ «Чебышёв» и IBM BlueGene/P.

Результаты замеров времени на СКИФе представлены на рис. 13–16. Большие размеры матрицы не позволяли произвести расчет с использованием одного процесса, поэтому нормировка производилась на время работы программы на 4 процессах.

Видно, что при сохранении результатов в ходе работы программы, для матриц размером 512 и 1024 при увеличении числа процессов от 8 до 32 ускорение падает.

Эффективность работы программы при сохранении результатов в ходе работы уменьшается, относительно эффективности работы программы не сохраняющей результаты расчетов в файлы. Увеличение числа процессов эффективно при размере матрицы более 1024, если происходит сохранение результатов.

Результаты замеров времени работы алгоритма с использованием неявной схемы на кластере IBM Bluegene/P приведены на рис. 17–20.

Большие размеры матрицы не позволяли произвести расчет с использованием одного процесса, поэтому нормировка производилась на время работы программы на 8

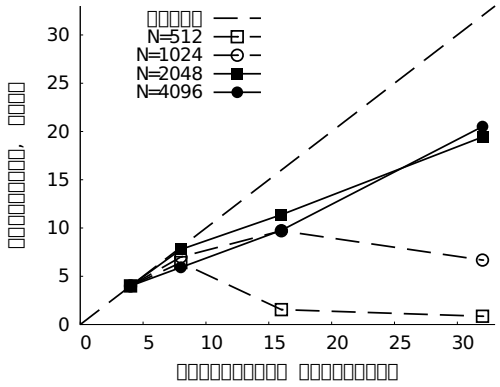

Рис. 17. Ускорение параллельного алгоритма с неявной схемой. BlueGene/P. Сохранение данных в файл не производилось.

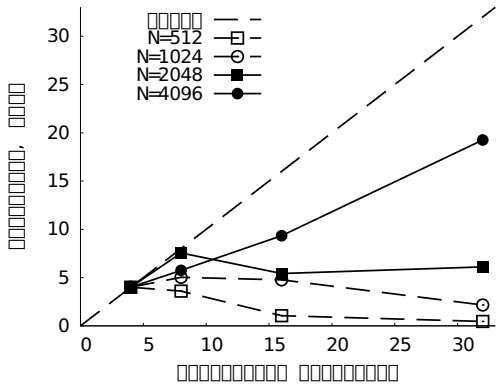

Рис. 18. Ускорение параллельного алгоритма с неявной схемой. BlueGene/P. Сохранение данных в файл производилось на каждом десятом шаге.

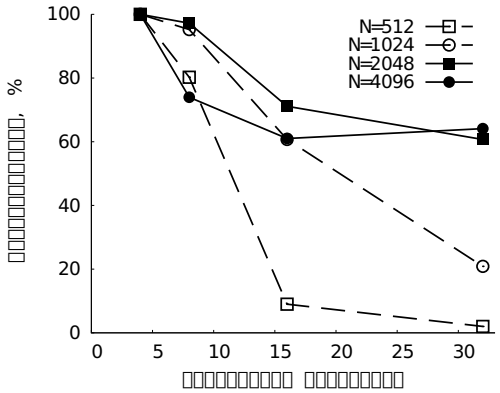

Рис. 19. Эффективность параллельного алгоритма с неявной схемой. BlueGene/P. Сохранение данных в файл не производилось.

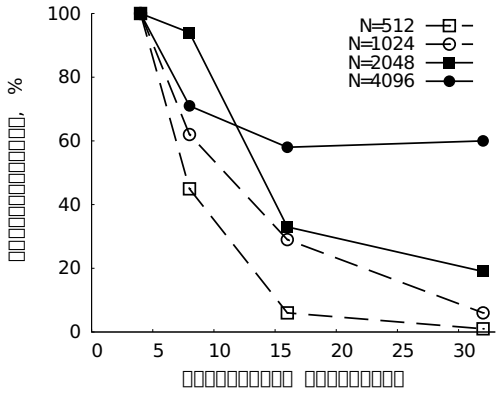

Рис. 20. Эффективность параллельного алгоритма с неявной схемой. BlueGene/P. Сохранение данных в файл производилось на каждом десятом шаге.

процессах. Видно стабильное линейное увеличение ускорения работы программы при увеличении числа процессоров уже для сравнительно небольших матриц. Эффективность реализации близка к единице. Таким образом, при работе на IBM Bluegene/P, увеличение числа процессов, на которых запускается программа, эффективно. Однако, время выполнения шага интегрирования на IBM Bluegene/P существенно больше, чем на СКИФе, что связано с существенно более слабыми процессорами.

5. Выводы

В рамках проектной работы по курсу «Суперкомпьютерные технологии» нами было проведено исследование работы различных алгоритмов решения нелинейного уравнения квазиоптики, известного также как нелинейное уравнение Шредингера. Рассматривались методы с использованием разностных схем и гибридный метод с расщепление по физическим факторам, где нелинейность учитывалась локально, а для решения линейной части уравнения использовался метод с применением преобразования Фурье.

Тестовые эксперименты показали, что при использовании схемы предиктор-корректор (Рунге-Кутта 4-го порядка) из-за наличия поперечных координат и участия в уравне-

нии производной по ним необходимо использовать довольно маленький шаг по координате z для получения устойчивого решения. Таким образом, эту схему имеет смысл использовать в расчётах, где величина шага уже лимитирована другими особенностями поставленной задачи. Также следует отметить, что для реализации этого алгоритма необходимо держать в памяти 3 временные матрицы равные по размеру основной, а на каждом шаге необходимо 4 раза обменяться граничными значениями локальных блоков матрицы.

В случае применения неявной консервативной схемы, устойчивой при любом шаге интегрирования, шаг необходимо выбирать основываясь на эмпирических физических оценках. Кроме того, для реализации этого алгоритма требуется только одна дополнительная матрица, равная основной, для хранения прогоночных коэффициентов. Алгоритм показал отличную масштабируемость на кластера IBM Bluegene/P, которая не сильно пострадала даже в случае периодического сохранения данных вычислений на диск. В основном это объясняется небольшой тактовой частотой процессоров при наличии быстрой сети и применения MPI Parallel I/O.

При использовании Фурье-метода можно обойтись без использования дополнительной матрицы, тем самым по сравнению с методом Рунге-Кутта увеличить возможной расчётной сетки в 2 раза по каждой координате. Из-за более сложной организации пересылок при расчёте параллельного Фурье-преобразования по сравнению с остальными алгоритмами его масштабируемость ниже. Однако скорость вычислений для этого метода больше, по крайней мере при использовании до 64 процессов, что для применения на СКИФ МГУ «Чебышёв» делает его более удачным.

Таким образом, нельзя чётко сказать, какой метод является лучше в общем случае. В случае, если задача не накладывает каких-то особых ограничений, лучше использовать Фурье-метод. Если же имеется возможность использовать для расчёта очень большое число процессоров, то возможно применения метода с неявной разностной схемой. Также этот метод будет применим для задач с неравномерной сеткой по поперечному сечению, так как для них нет метода быстрого преобразования Фурье и этот метод теряет свою актуальность.

6. Благодарности

- Администрацию СКИФ МГУ «Чебышёв» за терпимость к ночным нагрузкам
- Администрацию кластера IBM Bluegene/P ВМиК МГУ за предоставленные приоритеты
- Разработчиков FFTW за библиотеку быстрого преобразования Фурье
- Дональду Кнуту за создание TEX и Лесли Лэмпорту за создание LATEX.
- Участников сообщества http://xkcd.ru за добротный перевод комиксов

Исходный код работы(программы и все документы) можно скачать по адресу http://github.com/Sannis/msu_supercomputing_courcework/.

Эта версия отчёта будет считается финальной и изменению не подлежит. Дальнейшее развитие будет в статье, см. папку docs/results/paper_num_meth проекта. Олег Ефимов, 6.04.2010.

Список литературы

1. Пилипецкий Н.Ф., Рустамов А.Р. Письма в ЖЭТФ, 2, 1965.
2. Кандидов В.П., Шлёнов С.А., Косарева О.Г. *Филаментация мощного фемтосекундного лазерного излучения*, Квантовая электроника, 39, 3, 2009.
3. Ахманов С.А., Никитин С.Ю. *Физическая оптика*, М., Наука, 2004.
4. http://fftw.org
5. V.P. Kandidov, V.Yu. Fedorov *Properties of self-focusing in elliptic beams*, Quantum Electronics, 34, 12, 2004.