Раздел VI. Параллельные алгоритмы

УДК 551.466

А.Е. Чистяков

ТЕОРЕТИЧЕСКИЕ ОЦЕНКИ УСКОРЕНИЯ И ЭФФЕКТИВНОСТИ ПАРАЛЛЕЛЬНОЙ РЕАЛИЗАЦИИ ПТМ СКОРЕЙШЕГО СПУСКА

Рассматривается параллельная реализация алгоритма ПТМ скорейшего спуска для решения сеточных уравнений. Приведены оценки ускорения и эффективности работы параллельного алгоритма в случаях декомпозиции по одному и двум пространственным направлениям. Получена зависимость ускорения от числа процессоров.

Параллельный алгоритм; попеременно треугольный метод; теоретические оценки ускорения.

A.E. Chistyakov

SPEEDUP AND EFFICIENCY ESTIMATION OF PARALLEL SSOR ALGORITHM

Parallel realization of analog SSOR has been presented at the paper. Speed-up and efficiency estimation for 1D and 2D domain decomposition techniques has been presented. The relation between speed-up and involved processors is given.

Parallel realization; SSOR; Estimates for speedup.

Параллельный алгоритм описан с учетом особенностей имеющегося в Технологическом институте Южного федерального университета (ТТИ ЮФУ) HPC-кластера, представляющего собой систему с распределенной памятью, из 128 узлов, объединенных сетью InfiniBand, с пропускной способностью 20 Гбит/с., каждый узел, в свою очередь, является SMP-системой с 16-ю вычислителями и объемом ОЗУ 32 Гбайта. Параллельные вычисления с применением технологий MPI производились на кластере распределенных вычислений с использованием 128 процессоров. Проведенные численные эксперименты показали, что максимальное ускорение (для задачи размерностью 351x251x14), равное 43,6, достигалось на 128 процессорах.

Для оценки характеристик параллельного алгоритма нам понадобятся некоторые фактические данные о производительности вычислительной системы. Данные получены в результате тестирования пакетами Linpack и Pallas MPI benchmark.

Фактическая пропускная способность и латентность сети Infiniband, для узлов в разных корзинах (худший случай):

- латентность: 1,82 мкс;
- пропускная способность: 1224.38 Гбайт/с.

Пиковая производительность одного вычислительного ядра узла кластера: 9.2 GFlops.

Метод решения сеточных уранений. В конечномерном гильбертовом пространстве H рассматривается задача об отыскании решения операторного уравнения

$$Ax = f, \quad A : H \to H, \tag{1}$$

где A – линейный, самосопряженный ($A = A^*$), положительно определенный оператор ($A > 0$). Для нахождения задачи (1) будем использовать неявный итерационный процесс

$$B\frac{x^{m+1} - x^m}{\tau} + Ax^m = f, \quad B : H \to H. \tag{2}$$

где m – номер итерации, $\tau > 0$ – итерационный параметр, а B – некоторый обратимый оператор. По определению обращения оператора B в (2) должно быть существенно проще, чем непосредственное упрощение исходного оператора A в (1). При построении B будет исходить из аддитивного разложения

$$A = A_1 + A_2, \quad A_1^* = A_2. \tag{3}$$

В силу (3) $(Ay, y) = 2(A_1 y, y) = 2(A_2 y, y)$. Поэтому в (3) $A_1 > 0, A_2 > 0$. Пусть в (2)

$$B = (E + \omega A_1)(E + \omega A_2), \omega > 0, y \in H. \tag{4}$$

Поскольку $A = A^* > 0$, то вместе с (3) это дает $B = B^* > 0$. Соотношения (2)-(4) задают попеременно-треугольный итерационный метод (ПТМ) [1] решения задачи (1).

Алгоритм попеременно-треугольного итерационного метода скорейшего спуска [5] для расчета сеточных уравнений имеет вид

$$r^m = Ax^m - f, \tag{5}$$

$$B(\omega_m)\omega^m = r^m, \tag{6}$$

$$\hat{\omega}_m = \frac{2\left\|\omega^m\right\|}{\left\|A\omega^m\right\|}, \tag{7}$$

$$\tau_{m+1} = \hat{\omega}_m + \frac{2\left\|\omega^m\right\|^2}{\left(A\omega^m, \omega^m\right)}, \tag{8}$$

$$x^{m+1} = x^m - \tau_{m+1}\omega^m, \tag{9}$$

$$\omega_{m+1} = \hat{\omega}_m. \tag{10}$$

Декомпозиция области по одному пространственному направлению. Каждый процессор получает область, полученную путем разбиения исходной области на части по одному из координатных направлений с пересечением в двух узлах (рис. 1).

После того как каждый процессор получит свою часть области, рассчитывается вектор невязки (5) и его равномерная норма (максимальный по модулю элемент) (рис. 2).

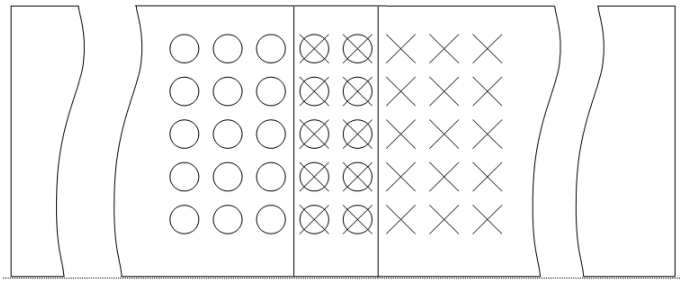

Рис. 1. Декомпозиция области. Кружками обозначены узлы одного процессора, крестами – соседнего к нему

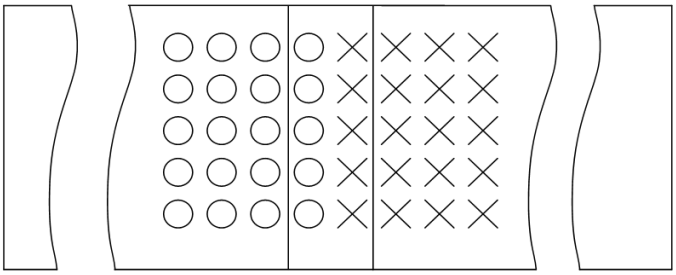

Рис. 2. Расчет вектора невязки

После чего каждый процессор определяет максимальный по модулю элемент и предает его значение каждому процессору (рис. 3).

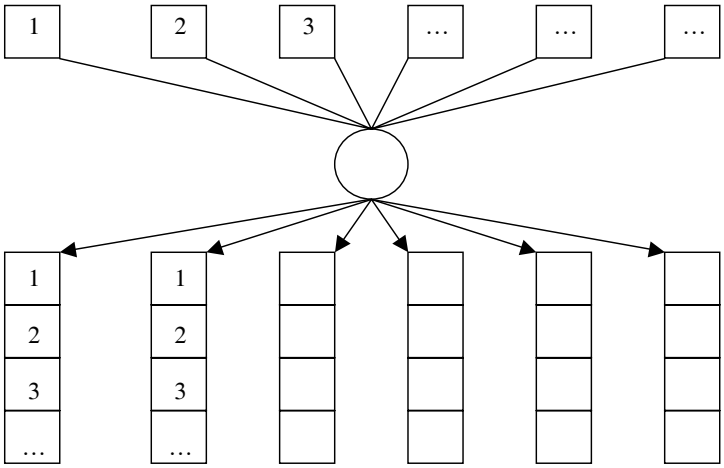

Рис. 3. Схема передачи от каждого процессора к каждому

После передачи каждый процессор считает максимальный элемент, в нем будет храниться норма вектора невязки.

Подставим (4) в уравнение (6)

$$(E + \omega_m A_1)(E + \omega_m A_2)\omega^m = r^m. \qquad (11)$$

Для определенности будем считать, что A_1 – нижнетреугольная матрица, тогда A_2 – верхнетреугольная. Задачу (11) можно разбить на две подзадачи

$$(E + \omega_m A_1) y^m = r^m, \tag{12}$$

$$(E + \omega_m A_2) \omega^m = y^m. \tag{13}$$

Вначале вычисляются сверху вниз вектор y^m, затем снизу вверх вектор ω^m. Схемы расчетов векторов y^m и ω^m изображены на рис. 4.

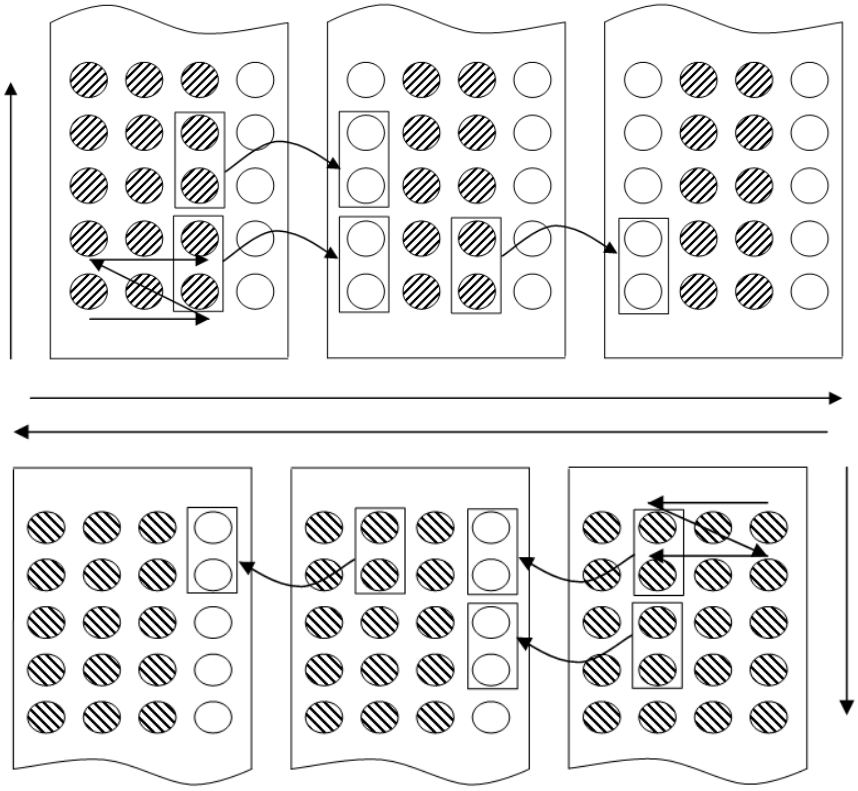

Рис. 4. Схема для расчета вектора y^m изображена сверху, вектора ω^m – снизу

В схеме для расчета вектора y^m только первый процессор не требует дополнительной информации и может независимо от других процессоров посчитать свою часть области, остальные процессоры ждет от предыдущего, пока он не передаст элементы, стоящие вначале строки. Передача по одному элементу не всегда оптимальна, так как появляются временные затраты связанные с организацией передачи, их можно избежать в случае увеличения объема передач. Данные рассуждения также используются для расчета вектора ω^m.

Далее считаются скалярные произведения

$$\left(\omega^m,\omega^m\right)=\sum_i\left(\omega^m\right)_i^2, \tag{14}$$

$$\left(A\omega^m,\omega^m\right)=\sum_i\left(A\omega^m\right)_i\left(\omega^m\right)_i, \tag{15}$$

$$\left(A\omega^m,A\omega^m\right)=\sum_i\left(A\omega^m\right)_i^2. \tag{16}$$

Каждый процессор считает свою часть скалярных произведений (14)-(16), затем осуществляется передача от всех ко всем рис 3. Значение скалярных произведений будет равно сумме всех передаваемых элементов. Остается только подставить значения скалярных произведений (14)-(16) в уравнения (7)-(8). С помощью формулы (9) осуществляется переход на следующую итерацию. Результаты использования многопроцессорных технологий для расчета полей течений приведены в табл. 1.

Таблица 1

Количество процессоров	Время, с,	Ускорение	эффективность
1	1447,415	1	1
2	795,307	1,82	0,91
3	598,875	2,42	0,81
4	504,043	2,87	0,71
5	399,695	3,62	0,72
6	352,892	4,1	0,68
7	304,012	4,76	0,68
8	287,366	5,04	0,63
9	270,897	5,34	0,59
10	262,657	5,51	0,55

Как видно из табл. 1 видно, что параллельный алгоритм попеременно-треугольного метода может быть применен для реальных задач и применение параллельных технологий вносит существенный вклад в увеличение скорости сходимости.

Декомпозиция области по двум пространственным направлениям. Каждый процессор получает область, полученную путем разбиения исходной расчетной области на части по двум координатным направлениям, рис. 5. Смежные области перекрываются в двух узлах по направлению, перпендикулярному плоскости разбиения.

После того как каждый процессор получит свою часть области, рассчитывается вектор невязки (5) и его равномерная норма (максимальный по модулю элемент). Расчет равномерной нормы вектора невязки каждый процессор определяет максимальный по модулю элемент и передает его значение каждому процессору. После передачи каждый процессор считает максимальный элемент, в нем будет храниться норма вектора невязки.

Вначале вычисляются сверху вниз вектор y^m, затем снизу вверх – вектор ω^m. Схема расчета вектора y^m изображена на рис. 6.

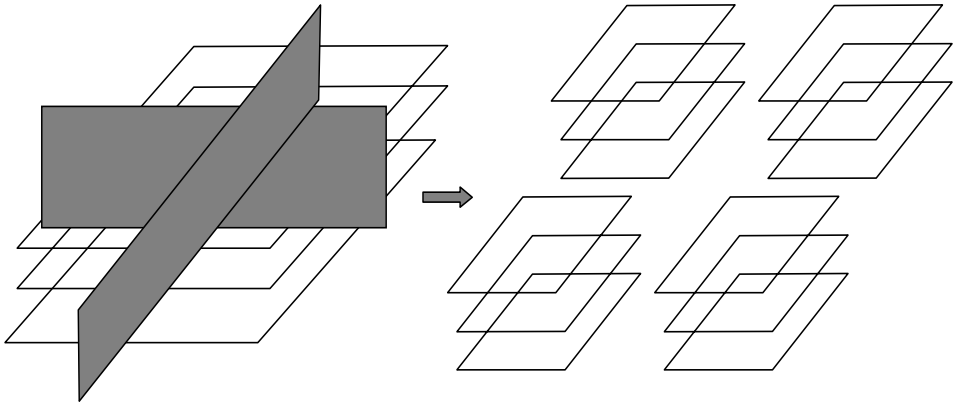

Рис. 5. Декомпозиция области

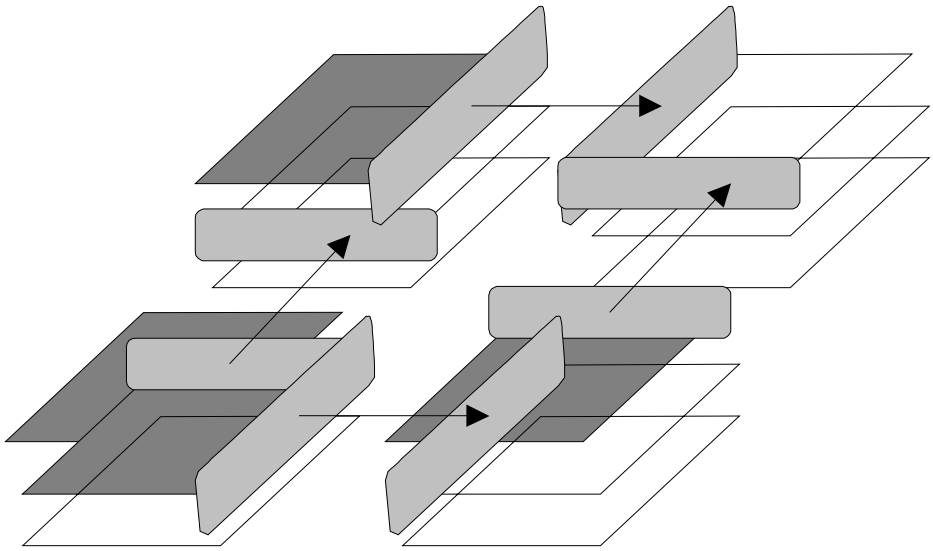

Рис. 6. Схема для расчета вектора y^m

Показана передача элементов после расчета двух слоев первым процессором.

На первом шаге вычислений первый процессор обрабатывает верхний слой. Затем осуществляется передача перекрывающихся элементов смежным процессорам. На следующем шаге первый процессор обрабатывает второй слой, а его соседи – первый. Передача элементов после расчета двух слоев первым процессором показана на рис 6. В схеме для расчета вектора y^m только первый процессор не требует дополнительной информации и может независимо от других процессоров вести обработку своей части области, остальные процессоры ждут результатов от предыдущего процессора, пока он не передаст вычисленные значения сеточных функций для узлов сетки, располагающихся в предшествующих позициях данной строки. Процесс продолжается до тех пор, пока не будут рассчитаны все слои. Далее считаются скалярные произведения (14)-(16).

Каждый процессор считает свою часть скалярных произведений (14)-(16), затем осуществляется передача от всех ко всем (рис. 3). Значение скалярных произведений будет равно сумме всех передаваемых элементов. Остается только подставить значения скалярных произведений (14)-(16) в уравнения (7)-(8). С помощью формулы (9) осуществляется переход на следующую итерацию. Результаты использования многопроцессорных технологий для расчета полей течений приведены в табл. 2.

Таблица 2

Количество процессоров	Время, с.	Ускорение	Эффективность
1	7,490639	1	1
2	4,151767	1,804	0,902
4	2,549591	2,938	0,734
8	1,450203	5,165	0,646
16	0,882420	8,489	0,531
32	0,458085	16,351	0,511
64	0,265781	28,192	0,44
128	0,171535	43,668	0,341

Из таблицы видно, что алгоритм попеременно-треугольного итерационного метода скорейшего спуска и его параллельная реализация на основе декомпозиции по двум пространственным направления могут эффективно применяться для решения задач гидродинамики при достаточно большом числе вычислительных узлов.

Теоретические расчеты ускорения и эффективности параллельной реализации ПТМ. Теоретически найдем трудоемкость, необходимую для выполнения шага ПТМ скорейшего спуска для СЛАУ с семидиагональной матрицей на последовательном вычислителе. Для расчета вектора невязки

$$r^m = Ax^m - f$$

нам необходимо выполнить 14N арифметических операций (сложение и умножение вместе).

Расчет вектора w^m, из уравнений

$$(E + \omega_m A_1) y^m = r^m,$$
$$(E + \omega_m A_2) w^m = y^m,$$

требуется по 9N операций на каждом полушаге (всего 18N), т.е. при расчете СЛАУ с верхнетреугольной и нижнетреугольной матрицей соответственно.

Для перехода на следующую итерацию

$$x^{m+1} = x^m - \tau_{m+1} w^m$$

потребуется еще 2N операции.

Также необходимо учесть количество арифметических операций затрачиваемых на расчет скалярных произведений в оптимизированном ПТМ скорейшего спуска

$$\left(w^m, w^m\right) = \sum_i \left(w^m\right)_i^2,$$

$$\left(Aw^m, w^m\right) = \sum_i \left(Aw^m\right)_i \left(w^m\right)_i,$$

$$\left(Aw^m, Aw^m\right) = \sum_i \left(Aw^m\right)_i^2,$$

а их еще 16N штук.

Итого общее количество арифметических операций, необходимых для осуществления одного шага ПТМ скорейшего спуска, составляет 50N.

Теоретически найдем время, необходимое для выполнения шага ПТМ скорейшего спуска, для СЛАУ с семидиагональной матрицей при помощи декомпозиции по одному пространственному направлению на кластере распределенных вычислений. На рис. 7 схематично изображены процессоры в кластере, их n штук. Вся наша область распределена между ними равномерно, т.е. каждый процессор получает область размером $\frac{N_x N_y N_z}{n}$, где N_x, N_y, N_z – количество узлов по пространственным направлениям. Так как данные между процессорами пересылаются не по одиночке, а пакетами, то целесообразно рассчитать объем оптимального пакета для передачи в условиях нашей задачи. Оптимальный объем пакета обозначим m.

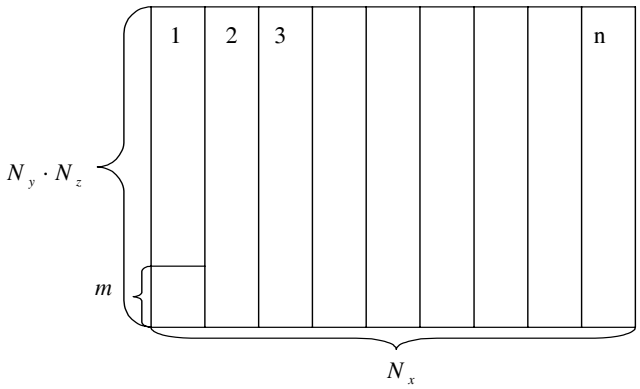

Рис. 7. Процессоры в кластере

Схема декомпозиции по одному пространственному направлению с пакетной организацией передачи.

На основании данных, полученных экспериментальным путем, были вычислены времена, затрачиваемые на так называемые накладные расходы.

А именно:

t_0 – время выполнения полезной операции;

t_x – время отклика (латентность);

t_n – время, затрачиваемое на передачу чисел с плавающей точкой.

Теперь получим выражение для расчета времени вычислительного процесса всеми процессорами.

Для расчета вектора невязки необходимо выполнить каждым процессором по $\frac{14N}{n}$ арифметических операций и сделать 2 передачи по $N_y N_z$ переменных. Таким образом, общее время расчета вектора невязки:

$$\frac{14N}{n}t_0 + 2\left(t_x + t_n N_y N_z\right).$$

Далее из-за малого количества передач при вычислении вектора невязки будем пренебрегать временем, затраченным на остановки счета.

Найдем время расчета вектора w^m.

$\frac{N_x}{n}$ – количество элементов в одной строке;

$\frac{N_x}{n}m$ – количество элементов в блоке (пакете);

$9\frac{N_x}{n}m \cdot t_0$ – время расчета одним процессором одного блока (пакета);

$n-1$ – количество передач, выполненное до старта последнего процессора;

$(n-1)t_x$ – время организации передачи при остановке счета;

$(n-1)t_n \cdot m$ – время, затраченное непосредственно на саму передачу одного блока (пакета);

$\left(9\frac{N_x}{n} * m \cdot t_0 + \left(t_x + m \cdot t_n\right)\right)(n-1)$ – общее время задержки старта, т.е. последний процессор начинает работу через данное время;

$\left(9\frac{N_x}{n} * m \cdot t_0 + \left(t_x + m \cdot t_n\right)\right)\frac{N_y N_z}{m}$ – время, затраченное на расчет вектора y^m последним процессором;

$\left(9\frac{N_x}{n} * m \cdot t_0 + \left(t_x + m \cdot t_n\right)\right)\left(\frac{N_y N_z}{m} + (n-1)\right)$ – время, затраченное на расчет вектора y^m;

$2\left(9\frac{N_x}{n} * m \cdot t_0 + \left(t_x + m \cdot t_n\right)\right)\left(\frac{N_y N_z}{m} + (n-1)\right)$ – время, затраченное на расчет вектора w^m.

Найдем оптимальное значение m

$$2\left(9\frac{N_x}{n} * m \cdot t_0 + \left(t_x + m \cdot t_n\right)\right)\left(\frac{N_y N_z}{m} + (n-1)\right) =$$

$$= 2\left(\left(9\frac{N_x}{n} \cdot t_0 + t_n\right)m + t_x\right)\left(\frac{N_y N_z}{m} + (n-1)\right) =$$

$$= 2\left(\left(9\frac{N_x}{n}\cdot t_0 + t_n\right)N_yN_z + t_x\frac{N_yN_z}{m} + \left(9\frac{N_x}{n}\cdot t_0 + t_n\right)m(n-1) + t_x(n-1)\right) =$$

$$= 18\frac{N}{n}\cdot t_0 + 2t_nN_yN_z + 2t_x(n-1) + 2t_x\frac{N_yN_z}{m} + \left(18\frac{N_x}{n}\cdot t_0 + 2t_n\right)(n-1)m =$$

$$= 18\frac{N}{n}\cdot t_0 + 2t_nN_yN_z + 2t_x(n-1) + \sqrt{2t_xN_yN_z\left(18\frac{N_x}{n}\cdot t_0 + 2t_n\right)(n-1)}$$

$$\left(\sqrt{\frac{2t_xN_yN_zn}{(18N_x\cdot t_0 + 2t_n)(n-1)}}\frac{1}{m} + \sqrt{\frac{(18N_x\cdot t_0 + 2t_n)(n-1)}{2t_xN_yN_zn}}m\right),$$

минимум достигается при

$$m = \sqrt{\frac{2t_xN_yN_zn}{(18N_x\cdot t_0 + 2t_n)(n-1)}}.$$

Время счета вектора w^m при оптимальном m равно

$$18\frac{N}{n}\cdot t_0 + 2t_nN_yN_z + 2t_x(n-1) + 4\sqrt{t_xN_yN_z\left(9\frac{N_x}{n}\cdot t_0 + 4t_n\right)(n-1)}.$$

$16\frac{N}{n}t_0$ – время, затраченное на расчет скалярных произведений скорейшего спуска. Временные затраты для перехода на следующую итерацию равны $2\frac{N}{n}t_0$.

Таким образом мы получили выражение для расчета времени вычислений на n процессорах:

$$t_N \approx 50\frac{N}{n}\cdot t_0 + 4t_nN_yN_z + 2t_xn + 4\sqrt{t_xN_yN_z\left(9\frac{N_x}{n}\cdot t_0 + 4t_n\right)(n-1)}$$

Теперь получим теоретические оценки ускорения и эффективности параллельной реализации ПТМ скорейшего спуска.

$A_{уск} = \frac{t_1}{t_N}$ – ускорение, отношение времени выполнения программы на одном процессоре, ко времени выполнения программы на n процессорах.

$t_1 = 50\cdot N\cdot t_0$;

$$t_N = 50\frac{N}{n}\cdot t_0 + 4t_nN_yN_z + 2t_xn + 4\sqrt{t_xN_yN_z\left(9\frac{N_x}{n}\cdot t_0 + 4t_n\right)(n-1)};$$

Тогда

$$A_{уск} = \frac{50Nt_0}{50\frac{N}{n}\cdot t_0 + 4t_n N_y N_z + 2t_x n + 4\sqrt{t_x N_y N_z\left(9\frac{N_x}{n}\cdot t_0 + 4t_n\right)(n-1)}},$$

$$A_{уск} = \frac{n}{1+\frac{2n}{25Nt_0}\left(t_n N_y N_z + \sqrt{t_x N_y N_z\left(9\frac{N_x}{n}\cdot t_0 + 4t_n\right)(n-1)}\right) + \frac{t_x n^2}{2525Nt_0}}.$$

$$E_{эф} = \frac{A_{уск}}{n} = \frac{1}{1+\frac{2n}{25Nt_0}\left(t_n N_y N_z + \sqrt{t_x N_y N_z\left(9\frac{N_x}{n}\cdot t_0 + 4t_n\right)(n-1)}\right) + \frac{t_x n^2}{2525Nt_0}}.$$

Таким образом, мы получили теоретической оценки эффективности и ускорения.

Теоретически найдем время, необходимое для выполнения шага ПТМ скорейшего спуска для СЛАУ с семидиагональной матрицей при помощи декомпозиции по двум пространственным направлениям на кластере распределенных вычислений. Вся наша область распределена между процессорами их n штук $n = n_x \cdot n_y$ (пусть $n_x \geq n_y$), т.е. каждый из них получает область размером $\frac{N_x N_y N_z}{n}$, где N_x, N_y, N_z количество узлов по пространственным направлениям.

Для расчета вектора невязки необходимо выполнить каждым процессором по $\frac{14N}{n}$ арифметических операций, а также сделать 2 передачи по $N_y N_z$ переменных и 2 передачи по $N_x N_z$. Таким образом, общее время расчета вектора невязки:

$$\frac{14N}{n}t_0 + 2t_n N_y N_z + 2t_n N_x N_z.$$

Найдем время расчета вектора w^m.

$\frac{N_x}{n_x}\frac{N_y}{n_y}$ – количество элементов в одном слое,

$9\frac{N_x N_y}{n}\cdot t_0$ – время расчета одним процессором каждого слоя,

$n_x + n_y - 2$ – количество передач, выполненных до запуска последнего процессора по $\frac{N_x}{n_x} + \frac{N_y}{n_y}$ элементов,

$\left(n_x + n_y - 2\right)\left(9\frac{N_x N_y}{n}t_0 + \left(\frac{N_x}{n_x} + \frac{N_y}{n_y}\right)t_n + t_x\right)$ – время задержки запуска последнего процессора,

$9\frac{N}{n}\cdot t_0$ – время счета последним процессором,

$\left(t_n N_y + t_x n_y\right)N_z + \left(t_n N_x + t_x n_x\right)N_z$ – общее время, затраченное на передачи при расчете вектора y^m;

Таким образом время расчета вектора w^m равно

$$18\frac{N}{n}\cdot t_0 + 2\left(t_n N_y + t_x n_y\right)N_z + 2\left(t_n N_x + t_x n_x\right)N_z +$$
$$+2\left(n_x + n_y - 2\right)\left(9\frac{N_x N_y}{n}t_0 + \left(\frac{N_x}{n_x} + \frac{N_y}{n_y}\right)t_n + t_x\right)$$

$16\frac{N}{n}t_0$ – время, затраченное на расчет скалярных произведений скорейшего спуска. Временные затраты для перехода на следующую итерацию равны $2\frac{N}{n}t_0$.

Таким образом, мы получили выражение для расчета времени вычислений на n процессорах:

$$50\frac{N}{n}\cdot t_0 + 4\left(N_x + N_y\right)N_z t_n + 2\left(n_x + n_y\right)t_x N_z +$$
$$+2\left(n_x + n_y - 2\right)\left(9\frac{N_x N_y}{n}t_0 + \left(\frac{N_x}{n_x} + \frac{N_y}{n_y}\right)t_n + t_x\right).$$

Рассмотрим случай $n_x = n_y = \sqrt{n}$, при этом время счета будет

$$t_N \approx 50\frac{N}{n}\cdot t_0 + 36\left(\sqrt{n} - 1\right)\frac{N_x N_y}{n}t_0 + 4N_z\left(t_n\left(N_x + N_y\right) + t_x\sqrt{n}\right).$$

Теперь получим теоретические оценки ускорения и эффективности параллельной реализации ПТМ скорейшего спуска.

$$A_{уск} = \frac{t_1}{t_N} = \frac{50Nt_0}{50\frac{N}{n}\cdot t_0 + 36\left(\sqrt{n} - 1\right)\frac{N_x N_y}{n}t_0 + 4N_z\left(t_n\left(N_x + N_y\right) + t_x\sqrt{n}\right)},$$

$$A_{уск} = \frac{n}{1 + \frac{36}{50N_z}\left(\sqrt{n} - 1\right) + \frac{4n}{50t_0}\left(t_n\left(\frac{1}{N_x} + \frac{1}{N_y}\right) + \frac{t_x\sqrt{n}}{N_x N_y}\right)},$$

$$E_{эф} = \frac{A_{уск}}{n} = \frac{1}{1 + \frac{36}{50N_z}\left(\sqrt{n} - 1\right) + \frac{4n}{50t_0}\left(t_n\left(\frac{1}{N_x} + \frac{1}{N_y}\right) + \frac{t_x\sqrt{n}}{N_x N_y}\right)}.$$

На рис. 8 изображены графики зависимости ускорения от количества процессоров, рассчитанные теоретически и экспериментально.

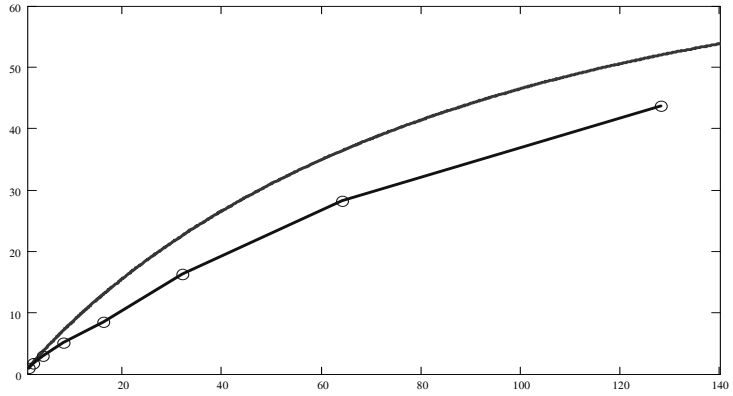

Рис. 8. Зависимость ускорения от числа процессоров. Сплошная кривая – зависимость, полученная теоретически, ломаная – экспериментально

В теоретических оценках ускорения рассматривается случай модельной задачи с прямоугольной областью. При решении задачи для реального водоема расчетная область имеет сложную форму. При этом реальное ускорение меньше его теоретической оценки. Из рис. 8 видно, что обе кривые проходят достаточно близко друг к другу, т.е. зависимость ускорения, полученную при теоретической оценки можно использовать в качестве оценки сверху ускорения для параллельной реализации алгоритма ПТМ скорейшего спуска путем декомпозиции области по двум пространственным направлениям.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Самарский А.А.* Теория разностных схем. – М.: Наука, 1989.
2. *Коновалов А.Н.* К теории попеременно-треугольного итерационного метода // Сибирский математический журнал. – 2002. – № 43 (3). – С. 552-572.

Чистяков Александр Евгеньевич
Технологический институт федерального государственного образовательного учреждения высшего профессионального образования «Южный федеральный университет» в г. Таганроге.
E-mail: cheese_05@mail.ru.
347928, Таганрог, пер. Некрасовский, 44.
Тел.: 88634371606.

Chistyakov Alexander Evgenjevich
Taganrog Institute of Technology – Federal State-Owned Educational Establishment of Higher Vocational Education “Southern Federal University”.
E-mail: cheese_05@mail.ru.
44, Nekrasovskiy, Taganrog, 347928, Russia.
Phone: +78634371606.