УДК 519.6

А.Н. ХИМИЧ, А.В. ПОПОВ, В.В. ПОЛЯНКО

АЛГОРИТМЫ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ ДЛЯ ЗАДАЧ ЛИНЕЙНОЙ АЛГЕБРЫ С МАТРИЦАМИ НЕРЕГУЛЯРНОЙ СТРУКТУРЫ

Ключевые слова: *линейная алгебра, разреженные симметричные матрицы, параллельные алгоритмы, эффективность.*

ВВЕДЕНИЕ

При численном решении научно-технических задач часто возникает необходимость в решении задачи (или нескольких задач) линейной алгебры — системы линейных алгебраических уравнений (СЛАУ) или частичной в общем случае обобщенной алгебраической проблемы собственных значений (АПСЗ). Как правило, для решения задач линейной алгебры требуется не менее 50 % времени решения всей задачи в целом. Например, задачи линейной алгебры возникают при дискретизации краевых задач или задач на собственные значения проекционно-разностным методом (конечных разностей, конечных элементов). Также при использовании итерационных методов решения нелинейных задач часто на каждой итерации решается линеаризованная задача — СЛАУ.

Важно отметить, что для задач линейной алгебры, возникающих при дискретизации, количество ненулевых элементов матриц составляет kn, где $k << n$, а n — порядок матрицы, т.е. матрицы являются разреженными. Структура разреженной матрицы определяется нумерацией неизвестных задачи и часто может быть ленточной, блочно-диагональной с окаймлением, профильной и т.д. Во многих случаях матрицы дискретных задач симметричны и положительно определены или положительно полуопределены.

Другой особенностью является высокий порядок матриц задач — до десятков миллионов. Это вызвано целесообразностью оперирования более точными дискретными моделями, что позволяет получать приближенные решения, более близкие к решениям исходных задач, учитывать локальные особенности данного процесса или явления.

Ввиду высоких порядков, несмотря на разреженную структуру матриц, решение таких задач требует значительных вычислительных ресурсов, которые могут быть предоставлены современными параллельными компьютерами, в частности компьютерами MIMD-архитектуры. При этом, однако, развитие вычислительной техники — появление многоядерных процессоров — диктует необходимость распараллеливания для эффективного использования высокопроизводительного вычислительного ресурса. Возникает проблема адаптации существующего программного обеспечения к новой архитектуре компьютеров. Поскольку во многих прикладных программных средствах можно выделить три части: препроцессор, процессор (решатель) и постпроцессор, то проблему адаптации можно достаточно быстро решить заменой решателя или отдельных его модулей соответствующими параллельными аналогами. Поэтому является актуальным создание для MIMD-компьютеров эффективных параллельных алгоритмов решения задач линейной алгебры с разреженными матрицами.

Под эффективным алгоритмом решения понимается алгоритм, который позволяет получить достоверное решение задачи с минимальным использованием ресурсов компьютера — процессоров, памяти, времени. Качество параллельных алгоритмов оценивается, как правило, коэффициентами ускорения и эффективности [1]. Важно также определить, какой алгоритм наиболее эффективен для решения конкретной задачи.

Эффективность параллельных алгоритмов в значительной степени зависит от сбалансированности загрузки процессоров, которая при решении задач линейной алгебры во многом определяется способами распределения между процессами, хранения и обработки матриц и векторов решаемой задачи. При этом необходимо стремиться к равномерной загрузке в каждый момент времени всех процессоров, участвующих в решении задачи, минимизируя время пребывания процессоров в состоянии ожидания. Также при разработке параллельных решателей для существующих программных средств необходимо учитывать структуру разреженных матриц, формируемых этими программными средствами во избежание дополнительных переформирований.

МЕТОДЫ РЕШЕНИЯ НЕКОТОРЫХ ЗАДАЧ ЛИНЕЙНОЙ АЛГЕБРЫ

Среди множества задач линейной алгебры с разреженными матрицами ключевое место занимает решение СЛАУ с симметричными положительно-определенными матрицами.

Рассмотрим решение СЛАУ $Ax = b$ с полуопределенной матрицей методом трехэтапной регуляризации [2, 3]. Этот метод состоит в последовательном выполнении следующих шагов:

1) последовательное решение двух СЛАУ: $(A+\alpha_0 I)z = b_k$ и $(A+\alpha_0 I)u = Az$ при произвольно выбранном параметре $\alpha_0 > 0$ (например, $\alpha_0 = 0{,}01$);

2) решение СЛАУ $(A+\alpha_0 I)w = u_H$ и вычисление $\mu = \max\limits_i |w_i|$, где $u_H \equiv \left(\max\limits_i |u_i|\right)^{-1} u$;

3) проверка выполнения неравенства $(2\alpha_0 + \|A\|\varepsilon_b)\mu \le \varepsilon$; если неравенство выполняется, то необходимая точность ε достигается при выбранном α_0; если неравенство не выполняется, то по формуле $\alpha_1 \le \frac{1}{2}\left(\frac{\varepsilon}{\mu} - \|A\|\varepsilon_b\right)$ определяется новое значение сдвига α_1 и повторяется первый шаг с матрицей $A+\alpha_1 I$ — вычисляется новое приближение u, которое обеспечивает необходимую точность ε нормального псевдорешения.

Таким образом, необходимо находить решения по меньшей мере трех СЛАУ с положительно-определенной матрицей. Кроме того, следует обратить внимание на вычисление произведения разреженной симметричной матрицы на вектор.

Для решения частичной обобщенной АПСЗ $Ax = \lambda Bx$ можно использовать метод итераций на подпространстве [2, 4], который заключается в построении последовательности подпространств E_t $(t = 1, 2, \dots)$, сходящейся к подпространству E_∞, которое содержит искомые собственные векторы. Для этого на t-й итерации вычисляется ортонормированный базис подпространства E_t и, если выполняются условия окончания итерационного процесса $\frac{|\lambda_i^{(t)} - \lambda_i^{(t-1)}|}{\lambda_i^{(t)}} \le \varepsilon$ $(i = 1, 2, \dots, r,$ а собственные значения упорядочены по возрастанию $0 < \lambda_1 \le \lambda_2 \le \dots \le \lambda_r \le \dots)$, определяются приближения к искомым собственным парам. Таким образом, для $t = 1, 2, \dots$ выполняются следующие операции:

1) решение СЛАУ $AX_t = Y_{t-1}$;

2) вычисление прямоугольной матрицы $W_t = BX_t$;

3) решение полной проблемы собственных значений $A_t Z_t = B_t Z_t \Lambda_t$ для проекций матриц A и B на подпространство E_t ($A_t = X_t^{\mathrm{T}} Y_{t-1} \equiv X_t^{\mathrm{T}} AX_t, B_t = X_t^{\mathrm{T}} W_t \equiv X_t^{\mathrm{T}} BX_t$);

4) вычисление $Y_t = W_t Z_t$;

5) проверка условий окончания итерационного процесса; если эти условия выполняются после t итераций, то в качестве приближенного решения задачи принимаются $\lambda_i^* = \lambda_i^{(t+1)}$, $X^* = X_{t+1} Z_{t+1}$ $(i = 1, 2, \ldots, r)$.

Таким образом, и в этом случае на каждой итерации решается СЛАУ с положительно-определенной матрицей. Если матрица B не является диагональной, на каждой итерации должно вычисляться произведение разреженной симметричной матрицы на плотную прямоугольную матрицу.

Кроме собственно решения задачи линейной алгебры, для гарантирования достоверности результатов необходимо проводить исследования математических свойств компьютерной модели задачи. Методы и алгоритмы таких исследований представлены в [2], где многие из них базируются на решении СЛАУ.

Также решение СЛАУ с разреженными симметричными положительно-определенными матрицами используется в алгоритмах решения нелинейных систем уравнений, при интегрировании систем дифференциальных уравнений и других задачах. Поэтому далее рассмотрим более подробно решение СЛАУ с разреженными симметричными положительно-определенными матрицами.

МЕТОДЫ РЕШЕНИЯ СЛАУ С РАЗРЕЖЕННЫМИ СИММЕТРИЧНЫМИ ПОЛОЖИТЕЛЬНО-ОПРЕДЕЛЕННЫМИ МАТРИЦАМИ

Проблема разработки эффективных методов решения СЛАУ с разреженными симметричными положительно-определенными матрицами является весьма актуальной, о чем свидетельствует большое число публикаций. Первые детальные исследования методов решения СЛАУ с разреженными матрицами начали проводиться с появлением и развитием ЭВМ. Ввиду ограниченных возможностей ЭВМ выпуска 50-70 годов прошлого века первоначально развитие получили итерационные методы. Однако эффективность итерационных методов существенно зависит от априорной информации о свойствах матрицы задачи. Кроме того, в случае многократного решения, как и в приведенных выше примерах, для СЛАУ с одной и той же матрицей и различными правыми частями прямые методы оказываются намного эффективнее. Поэтому по мере наращивания вычислительных ресурсов компьютеров все более широкое развитие получают алгоритмы прямых методов решения СЛАУ с разреженными матрицами.

В 80-е годы прошлого века был опубликован ряд фундаментальных работ, которые освещали основные достижения в этом направлении. В монографии [5] описано большинство основных методов решения СЛАУ с разреженными симметричными положительно-определенными матрицами больших размеров. Работа [6] посвящена рассмотрению основных методов решения алгебраических задач с разреженными матрицами и технологии их применения. Исследование прямых методов решения СЛАУ изложены в книге [7], а монография [8] посвящена проблемам реализации приведенных методов на параллельных компьютерах.

На основе накопленного материала был определен ряд эффективных алгоритмов решения СЛАУ с разреженными матрицами, способы хранения разреженных данных, а также различные алгоритмы оптимизации заполнения. Подавляющее большинство новых разработок базируется на работах [9–11], в которых

проведен широкий обзор современных алгоритмов решения СЛАУ с разреженными матрицами.

Рассмотрим систему линейных алгебраических уравнений

$$Ax = b \tag{1}$$

с симметричной положительно-определенной матрицей A порядка n. Наиболее эффективным прямым методом решения такой задачи является, как известно [12], метод Холецкого.

В варианте LDL^{T} метода Холецкого (в некоторых случаях целесообразно использовать вместо нижней треугольной матрицы L верхнюю треугольную матрицу $U \equiv L^{\mathrm{T}}$, тогда можно говорить о версии метода Холецкого $U^{\mathrm{T}}DU$) решение системы (1) состоит в решении следующих трех подзадач:

• треугольное разложение матрицы системы

$$A = LDL^{\mathrm{T}} \quad \text{или} \quad A = U^{\mathrm{T}}DU \tag{2}$$

(D — диагональная матрица);

• решение треугольной системы

$$Ly = b \ \text{ или } \ DUy = b; \tag{3}$$

• решение треугольной системы

$$L^{\mathrm{T}}x = D^{-1}y \ \text{ или } \ U^{\mathrm{T}}x = y. \tag{4}$$

Элементы LDL^{T}-разложения (2) вычисляются по формулам

$$l_{ik} = t_{ik} / d_k,\ k = 1, \ldots, i-1,\ d_i = a_{ii} - \sum_{k=1}^{i-1} l_{ik} t_{ik};$$

$$t_{ji} = a_{ji} - \sum_{k=1}^{i-1} l_{ik} t_{jk},\ j = i+1, \ldots, n,\ i = 1, 2, \ldots, n, \tag{5}$$

где l_{ik} — элемент матрицы L, d_i — диагональный элемент матрицы D. Учитывая, что $U \equiv L^{\mathrm{T}}$, можно легко получить аналогичные формулы для вычисления элементов $U^{\mathrm{T}}DU$-разложения (2) матрицы СЛАУ.

В случае разреженных симметричных матриц общий объем вычислений существенно сокращается, если вычисления по формулам (5) выполнять только с заведомо ненулевыми элементами разложения матрицы. Обозначим $Z_i(L)$ множество вторых индексов ненулевых поддиагональных элементов i-й строки нижней треугольной матрицы L. Тогда суммирование в формулах (5) проводится только для $k \in Z_i(L)$ или $k \in Z_i(L) \cap Z_j(L)$. Таким образом, элемент $l_{ij} = 0$, если $a_{ij} = 0$ и $Z_i(L) \cap Z_j(L) = \varnothing$.

Если в строках нижней треугольной матрицы L содержится много нерегулярно расположенных нулевых элементов, то при вычислениях по формулам (5) существенно возрастают накладные расходы на поиск соответствующих пар ненулевых элементов. В этом случае целесообразно использовать другой порядок вычислений элементов треугольного разложения, причем более рационально находить верхнюю треугольную матрицу U. Таким образом, элементы $U^{\mathrm{T}}DU$-разложения можно вычислить по следующим формулам (при $t_{ij}^{(0)} \equiv a_{ij}$):

$$d_i = t_{ii}^{(i-1)},\ u_{ik} = t_{ik}^{(i-1)} / d_i,\ k = i+1, \ldots, n, \tag{6}$$

$$t_{jk}^{(i)} = t_{jk}^{(i-1)} - u_{ik} t_{ij}^{(i-1)},\ k = j, \ldots, n,\ j = i+1, \ldots, n,\ i = 1, 2, \ldots, n. \tag{7}$$

Эти выражения легко получить из формул для вычисления элементов треугольного разложения алгоритмом Холецкого в виде внешних произведений.

Легко увидеть, что все нулевые элементы исходной матрицы, расположенные в строках левее или в столбцах выше первого ненулевого элемента, остаются нулевыми в соответствующем треугольном разложении. Поэтому эти элементы в вычислениях не участвуют, а первые ненулевые элементы (левые в строках или верхние в столбцах) определяют профиль разреженной матрицы. Формулы (5) или (6), (7) свидетельствуют, что профили исходной матрицы и ее треугольного разложения совпадают. При этом заполненность профиля треугольного разложения разреженной матрицы возрастает по сравнению с исходной разреженной матрицей, хотя некоторые внутренние элементы профиля треугольного разложения могут оставаться нулевыми. Заполненность профиля треугольного разложения разреженной матрицы зависит от связей между неизвестными системы и регулируется их нумерацией. Существуют (например, [5]) различные алгоритмы переупорядочения неизвестных, которые позволяют как уменьшить заполненность профиля, так и улучшить другие характеристики профиля треугольного разложения, влияющие на общее количество арифметических операций при решении СЛАУ.

Для решения треугольных систем (3) и (4) можно выписать формулы, аналогичные формулам (5) или (6), (7). Поскольку количество арифметических операций при решении систем с треугольными матрицами существенно меньше (например, для ленточных матриц в m раз, m — ширина ленты), чем при треугольном разложении матрицы исходной системы, то далее акцентируем внимание на параллельных алгоритмах треугольного разложения разреженной симметричной матрицы.

СТРУКТУРЫ РАЗРЕЖЕННЫХ МАТРИЦ И ФОРМАТЫ ЗАДАНИЯ ДАННЫХ

При разработке алгоритмов решения задач с разреженными матрицами особое значение имеет выбор способов задания и хранения ненулевых элементов. Эти способы определяются структурой (размещением ненулевых элементов) разреженной матрицы и требованиями алгоритма решения задачи с такой матрицей.

В общем случае каждый ненулевой элемент a_{ij} разреженной матрицы определяется своим значением и значениями индексов i и j. Одной из наиболее универсальных схем хранения разреженных матриц является разреженный строчный формат [10], который позволяет несколько уменьшить объем хранимой информации. В соответствии с этим значения ненулевых элементов матрицы и вторых индексов j хранятся в двух массивах: AN и JA. Используется также массив указателей IA, отмечающих позиции массивов AN и JA, с которых начинается описание очередной строки. Разреженный строчный формат представляет собой одну из наиболее используемых схем хранения разреженных матриц. Эта схема предъявляет невысокие требования к памяти и при этом оказывается чрезвычайно удобной для некоторых важных операций с разреженными матрицами: сложение, умножение, перестановка строк и т.п.

Ометим, что такие универсальные схемы хранения данных, как разреженный строчный формат, не учитывают специфики структуры (портрета) матрицы, которая имеет важное значение при разработке алгоритма решения задачи, и поэтому такие схемы не всегда удобны для решения СЛАУ. Современные компьютеры имеют сложную многоуровневую память. При выполнении вычислений они стремятся максимально использовать самую быструю память — регистры, кэш-память первого уровня, минимизируя количество обращений к медленной памяти. Однако в практических задачах часто ненулевые элементы строк треугольного разложения разреженной симметричной матрицы расположены ком-

пактно группами или составляют одну группу. Использование такой информации об особенностях структуры треугольного разложения позволяет упростить алгоритмы решения, ускорить доступ к элементам этой матрицы и выполнение операций над ними.

Если количество нулевых элементов внутри профиля треугольного разложения разреженной матрицы сравнительно невелико, то все внутренние элементы считаются ненулевыми. Матрицу такой структуры часто называют профильной. Профильный формат данных предусматривает хранение значений внутренних элементов профиля, а также число таких элементов в каждой строке (в явной или неявной форме). Если максимальное количество этих элементов в одной строке не более чем в 1,2 раза превышает их среднее количество, то практические расчеты показывают, что такую матрицу можно считать ленточной. Тогда такое максимальное количество ненулевых элементов в одной строке будет составлять ширину ленты треугольного разложения и полуширину ленты исходной матрицы. Ленточный формат предусматривает хранение значений элементов ленты матрицы и значения ширины ленты. Матрицы такой структуры часто получают при исходной нумерации неизвестных, а также при использовании для оптимизации структуры матрицы алгоритма фактор-деревьев или алгоритма Катхилла–Макки [5], которые обеспечивают концентрацию ненулевых элементов вблизи главной диагонали.

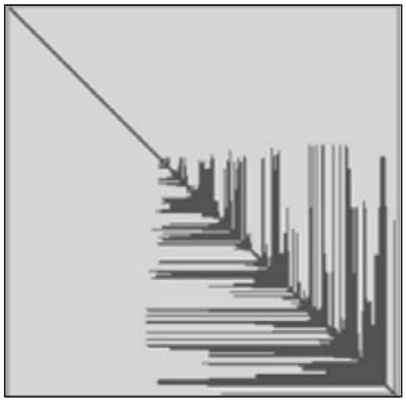

Рис. 1. Пример структуры небоскребной матрицы

Часто на практике, например при расчете конструкций, возникают задачи с симметричными матрицами особой структуры (рис. 1). В верхнем треугольнике таких матриц ненулевые значения размещаются вертикально по несколько столбцов подряд, по внешнему виду напоминая небоскребы (поэтому матрицы такой структуры иногда называют «небоскребными»). Таким образом, любая строка верхнего треугольника состоит из последовательных групп ненулевых и нулевых элементов. Поскольку ненулевые элементы сохраняются группами, то нет необходимости хранить индексы каждого элемента, что позволяет уменьшить размеры индексных массивов. Такую структуру матриц можно получить в случае использования алгоритма минимальной степени [5], который минимизирует количество ненулевых элементов в профиле треугольного разложения разреженной матрицы.

Схема задания ненулевых элементов группами является модификацией разреженного строчного формата для случая матриц описанного вида. Для задания каждой строки разреженной матрицы в этой схеме используются значения и количество ненулевых элементов матрицы в строке, количество групп ненулевых элементов матрицы в строке, количество элементов в каждой группе ненулевых или нулевых элементов. Таким образом, если группы ненулевых элементов включают три или более элементов, то в рассматриваемой схеме задается меньший объем данных, чем при использовании разреженного строчного формата.

РАСПРЕДЕЛЕНИЕ ДАННЫХ ПРИ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЯХ

Как отмечалось выше, эффективность параллельных алгоритмов в значительной степени зависит от сбалансированности загрузки процессов. При решении СЛАУ решающую роль в балансировании загрузки процессов, учитывая об-

щее количество арифметических операций, играет распределение данных между процессами при выполнении треугольного разложения матрицы. При этом распределение между процессами элементов треугольного разложения разреженной матрицы определяет распределение всех остальных данных, участвующих в вычислениях. В зависимости от распределения данных можно получить ту или иную алгоритмическую реализацию определенного метода.

Наиболее простым из существующих распределений является одномерное блочное распределение элементов матрицы (рис. 2, *а*). В этом случае каждый процесс содержит только один блок строк матрицы. Строка k размещена в процессе с номером k / t_c, где $t_c = n / c$ — максимальное число строк, размещенных в процессе, c — количество процессов. Однако эта схема не дает достаточной сбалансированности загрузки процессов ввиду так называемого эффекта Гайдна. Он заключается в том, что как только первые t_c строк обрабатываются, 0-й процесс заканчивает работу, затем после обработки следующих t_c строк заканчивает работу первый процесс и т.д. При этом в случае разреженной матрицы процессы не только одновременно не заканчивают работу, но и не начинают ее.

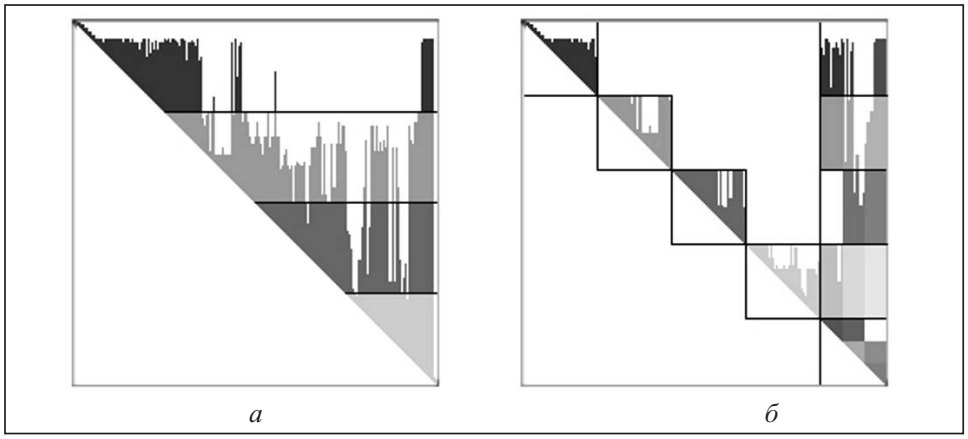

Рис. 2. Схема одномерного блочного (*а*) и блочно-диагонального с окаймлением (*б*) представления матрицы

Определенным выходом из такой ситуации послужило представление разреженной матрицы в виде блочно-диагональной с окаймлением [2, 13] (рис. 2, *б*). В этом случае количество диагональных блоков равно количеству используемых процессов. Такое представление можно получить, используя для оптимизации структуры матрицы алгоритм параллельных сечений [5]. Однако параллельные алгоритмы треугольного разложения матриц такой структуры достаточно эффективны только для сравнительно небольшого размера обрамления и малого количества процессов (до 20–30).

Значительным шагом к улучшению балансирования загрузки процессов, а следовательно и эффективности вычислительных алгоритмов, является циклический способ хранения и обработки матриц, что приводит к сбалансированной схеме алгоритма [13]. В соответствии со строчно-циклической схемой матрица распределяется между p процессами следующим образом: в i-м процессе размещаются строки с номерами $i, i+p, i+2p, \ldots, i+(p-1)p$. В результате достигается почти одинаковый объем вычислений в каждом процессе при реализации алгоритмов, т.е. практически исключается влияние эффекта Гайдна. Однако в строчно-циклическом способе отсутствие эффекта Гайдна «компенсируется» другим недостатком — слишком большим количеством синхронизаций между процессами. Тогда как и в блочной схеме обмены между процессами выполняются лишь

один раз за p шагов, строчно-циклическая схема требует пересылки данных при преобразовании каждой строки матрицы.

Компромиссом между одномерным блочным и строчно-циклическим представлениями является одномерное блочно-циклическое представление, при котором данные распределяются между процессами циклически, но блоками (рис. 3). Изменением размера блока можно получить как блочный, так и строчно-циклический вариант. Удачно подобранная величина размера блока позволяет практически устранить эффект Гайдна, при этом сохраняя количество синхронизаций на допустимом уровне.

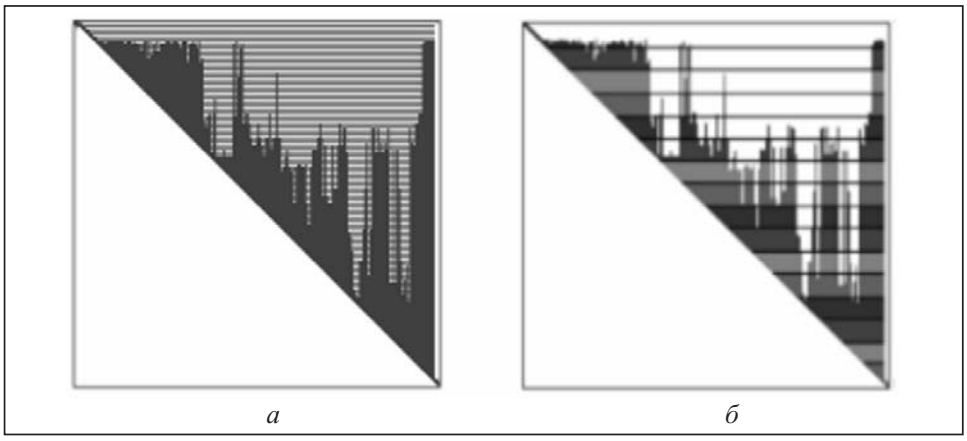

Рис. 3. Схема строчно-циклического (*а*) и одномерного блочно-циклического (*б*) представления матрицы

ПАРАЛЛЕЛЬНЫЕ АЛГОРИТМЫ ТРЕУГОЛЬНОГО РАЗЛОЖЕНИЯ РАЗРЕЖЕННЫХ СИММЕТРИЧНЫХ МАТРИЦ

Параллельные алгоритмы исследования и решения задач линейной алгебры с симметричными матрицами достаточно полно представлены в монографиях [2, 13]. Поэтому акцентируем внимание на параллельных алгоритмах треугольного разложения разреженной симметричной матрицы, отметив алгоритмы, которые не описаны в названных выше монографиях.

Исторически первые параллельные алгоритмы треугольного разложения разреженной симметричной матрицы были разработаны для блочно-диагонального с окаймлением представления такой матрицы [13, 14]. Достаточно подробно такой алгоритм описан также в работе [2] для решения задач с узкими ленточными матрицами. Однако возможности параллельных алгоритмов этой группы ограничены ввиду появления в процессе разложения приведенной матрицы, порядок которой равен количеству строк (столбцов) в окаймлении. Треугольное разложение такой матрицы может быть выполнено или в последовательном режиме (если ее порядок сравнительно небольшой), или в параллельном, но для этого необходимо перераспределить данные между процессами и использовать параллельный алгоритм для плотных симметричных матриц.

В [15, 16] рассмотрен строчно-циклический параллельный алгоритм метода Холецкого для ленточных симметричных матриц. В дальнейшем на базе этого алгоритма был разработан одномерный блочный циклический алгоритм метода Холецкого для ленточных и профильных симметричных матриц (например, [2]).

Одномерные блочные циклические параллельные алгоритмы LDL^{T}-разложения ленточной и профильной симметричных матриц практически совпадают, но вычисления по формулам (5) проводятся только для элементов соответствен-

но ленты или профиля матрицы. Итак, для каждого $I = 0, s, 2s, \ldots, N$, $N = s\lceil (n-1)/s \rceil$ (здесь s — число строк в блоке и значение $\lceil a \rceil$ равно целой части числа a) выполняется следующая последовательность операций:

1) при $I > 0$ каждым процессом согласно (5) вычисляются значения входящих в ленту или профиль элементов матрицы t_{ij} $(j = I - s + 1, \ldots, I;\ i = I + 1, \ldots, n)$ в соответствии со схемой распределения элементов исходной матрицы A;

2) завершается вычисление ведущим процессом элементов l_{ik} и d_i $(k = 1, \ldots, i-1;\ i = I + 1, \ldots, \min\{I + s, n\})$, входящих соответственно в ленту или профиль матрицы ведущего блока матриц по формулам (5);

3) с помощью операции мультирассылки вычисленный ведущим процессом массив элементов l_{ik} и d_i $(k = 1, \ldots, i-1;\ i = I + 1, \ldots, \min\{I + s, n\})$ ведущего блока рассылается всем процессам;

4) проверка выполнения условий положительной определенности $(d_i > 0)$ или невырожденности $(|d_i| > \texttt{macheps}||A||)$ исходной матрицы A одновременно всеми процессами для $i = I + 1, \ldots, \min\{I + s, n\}$.

В этом алгоритме вычисления можно проводить только с заведомо ненулевыми элементами матриц (исходной и треугольного разложения), используя, например, схему задания группами таких элементов. Однако накладные расходы на поиск таких элементов могут превысить выигрыш от сокращения количества арифметических операций.

Эффективное использование одномерных блочных алгоритмов возможно при хорошей сбалансированности загрузки процессов, не допускающей простоев отдельных процессов при выполнении вычислений. В общем случае этого можно достичь для ленточных матриц. Если количество элементов, с которыми проводятся операции, в строках профиля матрицы существенно отличается, то вычисления могут потребовать такого же времени, как и в случае ленточной матрицы с шириной ленты, равной максимальному количеству элементов профиля в одной строке.

Для разреженных симметричных матриц нерегулярной структуры часто можно достичь лучшей сбалансированности, если оперировать верхней треугольной матрицей U и использовать $U^{\mathrm{T}}DU$-разложение матрицы. В этом случае при использовании схемы задания ненулевых элементов группами и одномерного блочного циклического или строчно-циклического распределения данных для матрицы U между процессами можно достичь более равномерной загрузки процессов. Ниже приведен одномерный блочный циклический параллельный алгоритм для этого случая.

Для каждого $I = 0, s, 2s, \ldots, N$, $N = s\lceil (n-1)/s \rceil$ (здесь, как и выше, s — число строк в блоке и значение $\lceil a \rceil$ равно целой части числа a) выполняется последовательность операций:

1) с помощью операции мультирассылки массив ненулевых элементов из (7) $t_{ik}^{(I)}$ $(k = I + 1, \ldots, n;\ i = I + 1, \ldots, \min\{I + s, n\})$ $(I/s + 1)$-го блока, который далее называется ведущим, рассылается всем процессам;

2) для $i = I + 1, \ldots, \min\{I + s, n\}$

- каждым процессом проводится проверка выполнения условий положительной определенности $(d_i > 0)$ или невырожденности $(|d_i| > \texttt{macheps}||A||)$ исходной матрицы A;
- каждым процессом по формулам (6) вычисляются значения $1/d_i$ и ненулевых элементов u_{ik} $(k = i + 1, \ldots, n)$, входящих в $(i - I)$-ю строку ведущего блока треугольного разложения;

- каждым процессом вычисляются значения ненулевых элементов $t_{jk}^{(i)}$ $(k = i+1, \ldots, n;\ j = i+1, \ldots, n)$ по формулам (7) ведущего блока и остальных блоков в соответствии со схемой распределения между процессами элементов исходной матрицы A.

ЭФФЕКТИВНОСТЬ АЛГОРИТМОВ

Для оценки качества параллельных алгоритмов используются такие критерии, как коэффициенты ускорения и эффективности, которые соответственно могут быть определены следующим образом:

$$S_p = T_1 / T_p, \tag{8}$$

$$E_p = S_p / p, \tag{9}$$

где p — количество процессов, используемых для решения задачи, T_p — время решения задачи на MIMD-компьютере с использованием p процессов, T_1 — время решения той же задачи на гипотетическом последовательном компьютере с быстродействием одного процессора и оперативной памятью, которая равна суммарной памяти, используемой p процессами.

Вычислим времена выполнения последовательного и параллельного алгоритмов для нахождения коэффициентов (8) и (9) (используя обозначения из [2])

$$T_1 = O_1 t, \tag{10}$$

$$T_p = O_p t + O_{\text{o}} t_{\text{o}} + O_{\text{c}} t_{\text{c}}, \tag{11}$$

где t — среднее время выполнения одной арифметической операции (сложение или умножение) с плавающей запятой, O_1 и O_p — количество таких арифметических операций, выполняемых одним процессом, для последовательного и параллельного алгоритмов соответственно, t_{o} — время, необходимое для обмена одним машинным словом между двумя процессами, O_{o} — количество таких обменов, выполняемых одним процессом, t_{c} — время, необходимое для синхронизации двух процессов, O_{c} — количество таких синхронизаций, выполняемых одним процессом.

Обозначим η_i количество ненулевых элементов в i-й строке верхней треугольной матрицы $U^{\text{T}}DU$-разложения разреженной симметричной матрицы. Тогда справедлива теорема, аналогичная теореме 2.1.2 из [5] для LL^{T}-разложения.

Теорема 1. Для LDL^{T}-разложения или $U^{\text{T}}DU$-разложения разреженной симметричной матрицы требуется $O_1 = \sum_{i=1}^{n} \eta_i^2$ арифметических операций с плавающей запятой.

Доказательство. Легко увидеть, что формулы (6), (7) могут быть получены из (5) и наоборот — меняется только порядок вычислений. Следовательно, количества арифметических операций при LDL^{T}-разложении и $U^{\text{T}}DU$-разложении разреженной симметричной матрицы совпадают. Поэтому подсчитаем количество этих операций при $U^{\text{T}}DU$-разложении.

На i-м ($i = 1, 2, \ldots, n$) шаге для преобразования ведущей строки согласно (6) необходимо выполнить одно деление и $\eta_i - 1$ умножение. А для преобразования согласно (7) элементов k-й из $\eta_i - 1$ строки, номера которых соответствуют индексам ненулевых элементов ведущей строки, — по $\eta_i - k$ умножений и сложе-

ний, где $k = 1, 2, \ldots, \eta_i - 1$. Таким образом, общее количество арифметических операций с плавающей запятой на i-м шаге составляет $\eta_i + (\eta_i - 1)\eta_i = \eta_i^2$. Отсюда имеем $O_1 = \sum_{i=1}^{n} \eta_i^2$. ■

Заметим, что для ленточной симметричной матрицы $\eta_i = \min\{m, n-i\} + 1$. Следовательно, для треугольного разложения (LDL^{T} или $U^{\mathrm{T}}DU$) требуется $O_1 = nm^2 + O(nm + m^2)$ арифметических операций с плавающей запятой.

Далее, для определения T_p из (8) необходимо оценить $O_p, O_{\mathrm{c}}, O_{\mathrm{o}}$ из (11). В идеальном случае $O_p = O_1 / p$. Однако это возможно при полной сбалансированности загрузки процессов на каждом шаге алгоритма. Например, в приведенных выше алгоритмах часть вычислений выполняется без распараллеливания — или только одним процессом, или все процессы выполняют одни и те же вычисления с одними и теми же данными.

В монографии [2] приведены оценки значений $O_p, O_{\mathrm{c}}, O_{\mathrm{o}}$ и коэффициентов ускорения и эффективности упомянутых выше алгоритмов LDL^{T}-разложения ленточных, профильных и блочно-диагональных с окаймлением симметричных матриц. Например, коэффициенты ускорения и эффективности одномерного блочного параллельного алгоритма LDL^{T}-разложения ленточной симметричной матрицы при $n \geq sp^2$ и $m \geq sp$ можно оценить как

$$S_p \approx p\left(1 + \frac{2(p-1)(s+1)}{m} + \frac{p\log_2 p}{m}\tau_{1b}\right)^{-1}, \; E_p = \frac{S_p}{p},$$

где m — ширина ленты, s — количество строк в блоке, $\tau_{1b} = \frac{t_{\mathrm{o}}}{t} + \frac{1}{sm}\frac{t_{\mathrm{c}}}{t}$, а блочного параллельного алгоритма LDL^{T}-разложения узкой ленточной симметричной матрицы как

$$S_p \approx \frac{p}{4}\left(1 + \frac{1}{2m} + \frac{p-1}{4n}\left(\frac{7p-18}{3}m + \frac{p}{2}\tau_1\right)\right)^{-1}, \; E_p = \frac{S_p}{p};$$

если $\frac{1}{2m} + \frac{p-1}{4n}\left(\frac{7p-18}{3}m + \frac{p}{2}\tau_1\right) << 1$, то

$$E_p \approx 0{,}25 - \frac{1}{8m} - \frac{p-1}{16n}\left(\frac{7p-18}{3}m + \frac{p}{2}\tau_1\right),$$

где $\tau_1 = \frac{t_{\mathrm{o}}}{t} + \frac{2}{m^2}\frac{t_{\mathrm{c}}}{t}$.

Более подробно остановимся на оценках $O_p, O_{\mathrm{c}}, O_{\mathrm{o}}$ для предложенного выше одномерного блочно-циклического алгоритма $U^{\mathrm{T}}DU$-разложения матрицы (в форме внешних произведений).

Теорема 2. Одномерный блочный циклический алгоритм $U^{\mathrm{T}}DU$-разложения разреженной матрицы требует выполнения не менее $O_p^0 = \frac{1}{p}\left(\sum_{i=1}^{n}\eta_i^2 + (p-1)\sum_{i=1}^{n}\eta_i\right)$ арифметических операций с плавающей запятой, т.е. $O_p \geq O_p^0$.

Доказательство. Все p процессов при $U^{\mathrm{T}}DU$-разложении разреженной симметричной матрицы одномерным блочным циклическим параллельным алгоритмом выполняют $O_1 + (p-1)O^{(+)}$ арифметических операций с плавающей за-

пятой, где $O^{(+)}$ — количество арифметических операций с плавающей запятой, которые выполняются без распараллеливания — или только одним процессом, или всеми процессами с одними и теми же данными. Эти преобразования проводятся по формулам (6) и (7) ненулевых элементов ведущего блока. Легко увидеть, что $O^{(+)}$ ограничено снизу величиной $\sum_{I=0}^{s\lceil (n-1)/s\rceil} \sum_{i=I+1}^{\min\{I+s,n\}} \eta_i = \sum_{i=1}^{n} \eta_i$ — количеством операций, выполняемых по преобразованиям (6). Эта величина увеличивается на количество операций в преобразованиях (7) ненулевых элементов строк ведущего блока, номера которых соответствуют индексам ненулевых элементов текущей ведущей строки. Следовательно, $O^{(+)} \geq \sum_{i=1}^{n} \eta_i$. В предположении полной сбалансированности загрузки процессов на каждом шаге алгоритма каждый процесс выполняет не менее $(O_1 + (p-1)O^{(+)})/p \geq O_p^0$ арифметических операций с плавающей запятой. В общем случае величина O_1/p заменяется суммой наибольших количеств операций, выполняемых одним процессом на каждом шаге цикла по I. Таким образом, $O_p \geq (O_1 + (p-1)O^{(+)})/p \geq O_p^0$. ■

Теорема 3. Если для мультирассылки ненулевых элементов ведущего блока используется алгоритм дерева, то количество обменов (синхронизаций) между процессами для одномерного блочно-циклического параллельного алгоритма $U^{\mathrm{T}}DU$-разложения разреженной матрицы оценивается величиной $O_{\mathrm{c}} \approx \frac{n}{s}\log_2 p$, а общее количество данных, которыми обмениваются процессы, составляет приблизительно $O_{\mathrm{o}} \approx \log_2 p \sum_{i=1}^{n} \eta_i$ двойных слов.

Доказательство. На каждом шаге цикла по I выполняется по одной операции мультирассылки массива из $\sum_{i=I+1}^{\min\{I+s,n\}} \eta_i$ ненулевых элементов ведущего блока. Если для этой операции используется алгоритм дерева [17], то при одной мультирассылке выполняется приблизительно $\log_2 p$ синхронизаций, а общее число данных, которыми процессы обмениваются, составляют приблизительно $\log_2 p \sum_{i=I+1}^{\min\{I+s,n\}} \eta_i$. Следовательно, общее количество синхронизаций оценивается величиной $O_{\mathrm{c}} \approx (n/s)\log_2 p$, при этом общее количество данных, которыми обмениваются процессы, составляет приблизительно $O_{\mathrm{o}} \approx \log_2 p \sum_{i=1}^{n} \eta_i$. ■

Таким образом, используя результаты теорем 2 и 3 и с учетом (11), имеем

$$T_p \geq \frac{t}{p}\left(\sum_{i=1}^{n} \eta_i^2 + (p-1)\sum_{i=1}^{n} \eta_i\right) + t_{\mathrm{c}} \log_2 p \frac{n}{s} + t_{\mathrm{o}} \log_2 p \sum_{i=1}^{n} \eta_i.$$

Следовательно, с учетом также результатов теоремы 1 и выражений (8)–(10) для коэффициента ускорения одномерного блочного циклического параллельного алгоритма $U^{\mathrm{T}}DU$-разложения разреженной симметричной матрицы справедлива оценка сверху

$$S_p \leq p\left(1 + \frac{p-1}{O_1}\sum_{i=1}^{n} \eta_i + \frac{p\log_2 p}{O_1}\tau_{1s}\right)^{-1}, \qquad (12)$$

а также оценка

$$E_p \le 1 - \left(\frac{p-1}{O_1} \sum_{i=1}^{n} \eta_i + \frac{p \log_2 p}{O_1} \tau_{1s} \right)$$

для коэффициента эффективности, где $\tau_{1s} = \frac{t_c}{t} \frac{n}{s} + \frac{t_o}{t} \sum_{i=1}^{n} \eta_i$.

АПРОБАЦИЯ АЛГОРИТМОВ

На основе предложенных параллельных алгоритмов разработано соответствующее программное обеспечение для среды параллельного программирования MPI. Получены решения ряда задач на кластере СКИТ и интеллектуальном параллельном компьютере Инпарком [18].

В работах [2, 15, 16, 18] приведены многочисленные результаты апробации предложенных параллельных алгоритмов. Ниже даны результаты тестирования одномерного блочного циклического параллельного алгоритма решения СЛАУ с разреженными матрицами, который использует $U^{\mathrm{T}}DU$-разложение матрицы.

В табл. 1 приведены результаты апробации одномерного блочного циклического параллельного алгоритма на решении нескольких СЛАУ с разреженными матрицами. При этом для каждой из задач проводилось решение с матрицей исходной (неоптимизированной) структуры и переупорядоченной (оптимизированной) структуры с помощью алгоритма минимальной степени — в таблице эти случаи различаются значениями ширины ленты и заполненности профиля матриц.

Таблица 1

Порядок системы	Значение ширины ленты	Заполненность, %	Время (мин) работы программы для процессов					
			1	8	16	32	64	128
44 436	4 475	21	2,17	0,38	0,30	0,22	0,20	0,18
44 436	37 580	2	0,95	0,17	0,12	0,12	0,12	0,08
283 031	19 530	7	32,00	5,95	3,27	2,30	2,03	2,05
283 031	281 341	1	9,67	2,20	1,05	0,68	0,67	0,58
661 590	34 242	5	98,80	17,60	10,08	6,57	5,45	5,22
661 590	541 257	1	56,20	14,67	7,12	3,80	2,57	2,10

Приведенные в таблице времена получены при размере блока $s = 1$. В отличие от плотных матриц, ленточного или профильного представления матриц в рассматриваемом случае оптимальный размер блока необходимо подбирать для каждой конкретной задачи в зависимости от размещения ненулевых элементов. Поэтому для сравнения производительности алгоритма на различных задачах использовался именно строчно-циклический вариант, который обеспечивает, как правило, наилучшую сбалансированность загрузки процессов.

На рис. 4 приведены зависимости ускорения и эффективности алгоритма от количества процессов для приведенных в табл. 1 двух задач: с матрицей исходной структуры (и) и с переупорядоченной структурой (п).

На рис. 5 приведены зависимости ускорения рассматриваемого алгоритма от количества процессов при различных порядках СЛАУ, приведенных в табл. 1 (с переупорядоченной матрицей).

Приведенные в табл. 1 и на рис. 4, 5 результаты демонстрируют зависимость характеристик алгоритма (времени решения, ускорения, эффективности) от сбалансированности загрузки процессов. При этом чем меньше порядок системы, тем труднее достичь сбалансированности при наращивании числа процессов (см. рис. 5).

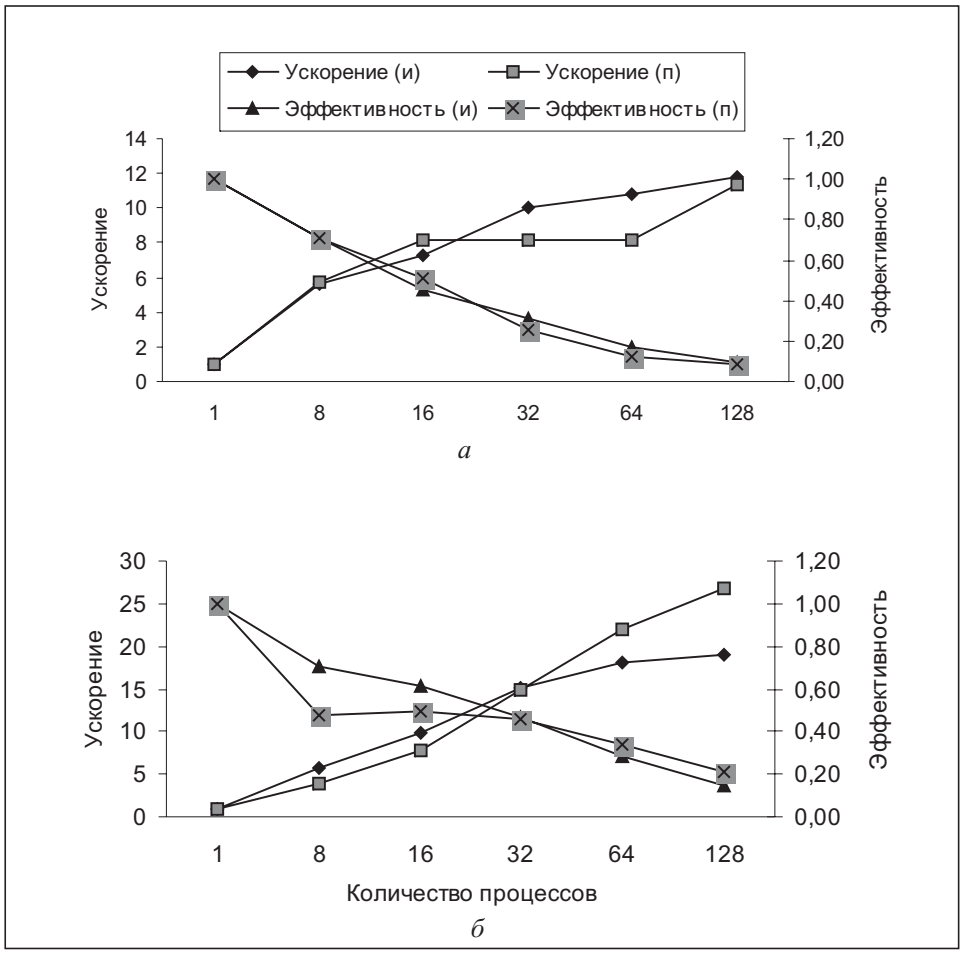

Рис 4. Зависимости ускорения и эффективности алгоритма от количества процессов для СЛАУ порядка 44 436 (*а*) и для СЛАУ порядка 661590 (*б*)

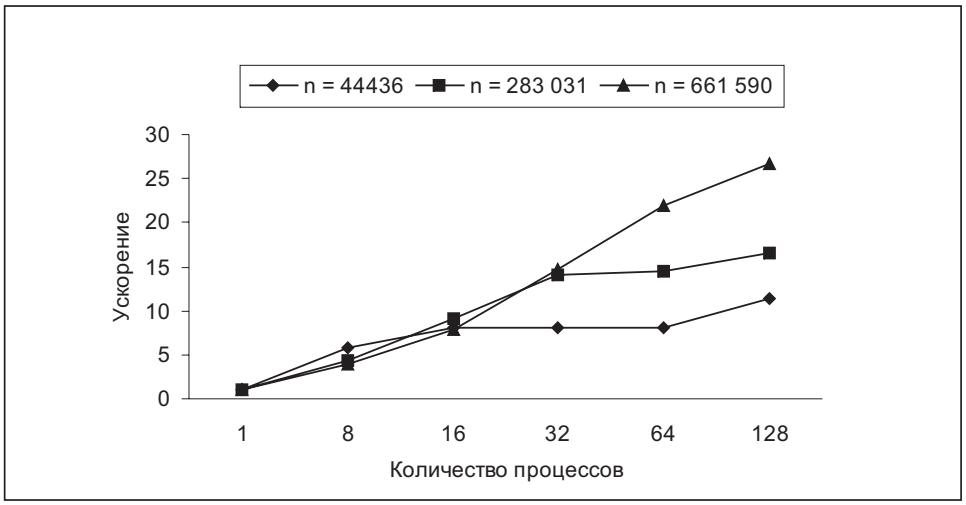

Рис. 5. Зависимости ускорения алгоритма от количества процессов для СЛАУ различных порядков

Следует также отметить, что в некоторых случаях при лучшей сбалансированности загрузки процессов могут быть достигнуты лучшие временные показатели при профильном представлении матрицы системы и использовании соот-

ветствующего параллельного алгоритма (одномерного блочного циклического). Так, используя 16 процессов для решения СЛАУ с исходной (непереупорядоченной) структурой матрицы порядка 44 436 потребовалось 0,16 мин, системы порядка 283 031 — 1,29 мин и для системы порядка 661 590 — 6,00 мин. Выбор наиболее эффективного параллельного алгоритма можно сделать лишь на основании экспериментальных данных, так как сбалансированность загрузки может существенно меняться при изменении количества процессов.

Приведем результаты решения двух задач большого объема. В табл. 2 даны временные характеристики решения СЛАУ с 5 371 727 неизвестными при расчете на прочность конструкции делового центра, состоящей из двух башен. Время решения этой задачи без распараллеливания составляет приблизительно 115 час. В табл. 3 приведены временные характеристики решения частичной обобщенной АПСЗ порядка 2 385 822 с использованием 247 процессов.

Таблица 2

Этапы алгоритма	Время (час:мин) решения (для процессов)						
	48	64	96	128	192	224	256
Чтение данных из файла	0:32	0:32	0:44	0:44	0:42	0:42	0:42
Треугольное разложение матрицы	9:14	8:22	6:24	5:12	4:38	4:19	4:07
Исследование свойств СЛАУ	0:23	0:20	0:13	0:09	0:06	0:05	0:04
Решение треугольных систем	0:11	0:10	0:06	0:04	0:03	0:02	0:02
Всего времени	10:22	9:24	7:27	6:09	5:29	5:08	4:55

Таблица 3

Этапы решения задачи	Время (час:мин) решения АПСЗ порядка 2 385 822	
	Профильное представление	Небоскребное представление
Чтение данных	0:38,2	0:06,2
Решение АПСЗ:		
Треугольное разложение матрицы	0:25,6	0:13,3
Итерационные процессы	0:03,9	0:18,9
Исследование достоверности	0:48,3	0:03,7
Сохранение результатов решения	1:35,5	1:22,3

Для профильного представления матрицы ширина ленты составляет 115480 час, заполненность — 3 %. Для небоскребного представления ширина ленты 2 051 632 час, заполненность — менее 1%).

При исследовании достоверности в случае небоскребного представления определялись только оценки вычислительной погрешности. Время решения этой задачи без распараллеливания превышает 15 час.

ЗАКЛЮЧЕНИЕ

Для задач с разреженными симметричными матрицами предложены различные параллельные алгоритмы, которые обеспечивают высокую эффективность распараллеливания, учитывают структуру разреженных матриц и их заполненность. Определены некоторые критерии выбора алгоритмов для решения конкретной задачи.

Для разреженных матриц нерегулярной структуры разработан и программно реализован одномерный блочный циклический параллельный алгоритм, в ряде случаев обеспечивающий лучшее балансирование процессов для разреженного

типа данных. Разработанный алгоритм позволяет распределить между процессами ненулевые элементы треугольного разложения разреженной матрицы таким образом, чтобы вычисления проводились одновременно большинством процессов. Предложены эффективные схемы распределения данных между процессами. Получены оценки сверху коэффициентов ускорения и эффективности рассматриваемых параллельных алгоритмов.

Дальнейшие исследования, направленные на поиск алгоритмов переупорядочения разреженных матриц, обеспечат высокую эффективность того или иного параллельного алгоритма решения задачи линейной алгебры в первую очередь за счет сбалансированности загрузки процессов.

СПИСОК ЛИТЕРАТУРЫ

1. Молчанов И.Н., Химич А.Н., Попов А.В. и др. Об эффективной реализации вычислительных алгоритмов на MIMD-компьютерах // Искусственный интеллект. — 2005. — № 3. — С. 175–184.
2. Химич А.Н., Молчанов И.Н., Попов А.В. и др. Параллельные алгоритмы решения задач вычислительной математики. — К.: Наук. думка, 2008. — 248 с.
3. Попов А.В., Химич А.Н. Исследование и решение первой основной задачи теории упругости // Компьютерная математика. — 2003. — № 2. — С. 105–114
4. Молчанов И.Н., Попов А.В., Химич А.Н. Алгоритм решения частичной проблемы собственных значений для больших ленточных матриц // Кибернетика и системный анализ. — 1992. — № 2. — С. 141–147.
5. Джордж А., Лю Дж. Численное решение больших разреженных систем уравнений. — М.: Мир, 1984. — 334 с.
6. Писанецки С. Технология разреженных матриц. — М.: Мир, 1988. — 410 с.
7. Эстербю О., Златев З. Прямые методы для разреженных матриц. — М.: Мир, 1987. — 118 с.
8. Ортега Дж. Введение в параллельные и векторные методы решения линейных систем. — М.: Мир, 1991. — 367 с.
9. Баландин М.Ю., Шурина Э.П. Методы решения СЛАУ большой размерности. — Новосибирск: Изд-во НГТУ, 2000. — 70 с.
10. Глушакова Т.Н., Эксаревская М.Е. Методы работы с разреженными матрицами произвольного типа. — Воронеж: Воронежский гос. ун-т, 2005. — 44 с.
11. Глушакова Т.Н., Блатова И.А. Методы решения систем с разреженными матрицами. — Воронеж: Воронежский гос. ун-т, 2000. — 36 с.
12. Уилкинсон Дж. Х., Райнш К. Справочник алгоритмов на языке Алгол. Линейная алгебра. — М.: Машиностроение, 1976. — 389 с.
13. Численные методы для многопроцессорного вычислительного комплекса ЕС / В.С. Михалевич, … , И.В. Сергиенко, …, А.Н. Химич и др. / Под ред. И.Н. Молчанова. — М.: Изд. ВВИА им. Н.Е. Жуковского, 1986. — 401 с.
14. http://www.netlib.org/scalapack
15. Сергиенко И.В., Дейнека В.С., Химич А.Н. и др. Исследование с помощью многопроцессорного вычислительного комплекса распределенных систем с большими объемами связанных данных. — Киев, 2005. — 33 с. — (Препр. / НАН Украины, Ин-т кибернетики им. В.М. Глушкова; 2005-1).
16. Попов А.В., Химич А.Н. Параллельный алгоритм решения системы линейных алгебраических уравнений с ленточной симметричной матрицей // Компьютерная математика. — 2005. — № 2. — С. 52–59.
17. http://www.mcs.anl.gov/mpi/mpich
18. Химич А.Н., Молчанов И.Н., Мова В.И. и др. Численное программное обеспечение MIMD–компьютера Инпарком. — Киев: Наук. думка, 2007. — 222 с.

Поступила 04.08.2010