УДК 519.67

OPENMP-РЕАЛИЗАЦИЯ БЛОЧНОГО АЛГОРИТМА ВЫЧИСЛЕНИЯ ПОПЕРЕЧНОЙИ ПРОДОЛЬНОЙ СКОРОСТЕЙ ТЕЧЕНИЯ КРОВИ В СОСУДЕ

Н. А. Лиходед[1)], Э. Е. Малиев[2)]

[1)] *Белорусский государственный университет,*
Минск, Беларусь, likhoded@bsu.by
[2)] *Институт математики НАН Беларуси,*
Минск, Беларусь, maliev@im.bas-net.by

Работа посвящена разработке и реализации на многоядерных системах с общей памятью алгоритма решения блочно-трехдиагональных систем линейных алгебраических уравнений с вложенной структурой блоков. Такие системы возникают при одновременном получении поперечной и продольной компонентов скорости движения крови в кровеносных сосудах. Используется красно-черное упорядочение модифицированного блочного метода Гаусса-Зейделя в неявной форме, обладающего естественным параллелизмом.

Ключевые слова: параллельные вычисления; многопроцессорная система с общей памятью; блочный метод Гаусса-Зейделя; красно-черное упорядочение; параллельные потоки вычислений.

OpenMP IMPLEMENTATION OF A BLOCK ALGORITHM FOR CALCULATING TRANSVERSE AND LONGITUDINAL BLOOD FLOW VELOCITIES IN A VESSEL

N. A. Likhoded[a)], E. E. Maliev[b)]

[a)] *Belarusian State University,*
Minsk, Belarus, likhoded@bsu.by
[b)] *Institute of Mathematics, National Academy of Science,*
Minsk, Belarus, maliev@im.bas-net.by

The paper is devoted to the development and implementation on multi-core shared-memory systems of an algorithm for solving block-tridiagonal systems of linear algebraic equations with a nested block structure. Such systems arise from the simultaneous determination of the transverse and longitudinal components of blood flow velocity in blood vessels. The red-black ordering of the modified Gauss-Seidel block method is used in an implicit form, which exhibits natural parallelism.

Keywords: parallel computing; shared-memory parallel computer; Gauss-Seidel block method; red-black ordering; parallel threads.

1. Введение

Настоящая работа во многом опирается на материал «Распределенный блочный алгоритм вычисления поперечной и продольной скоростей течения крови в сосуде» этого международного научного конгресса (см. с. 399–408). Здесь рассматриваются такие же блочно-трёхдиагональная система линейных алгебраических уравнений (СЛАУ) с трёхдиагональными диагональными матрицами-блоками вложенной структуры, метод решения, вспомогательные алгоритмы, направления дальнейших исследований, ссылки на литературу. Поэтому обозначения и всё перечисленное не будем дублировать, их надо смотреть в указанном материале. Новым в этой работе являются: алгоритмы решения СЛАУ, их ориентированность на многоядерные CPU, OpenMP-реализации алгоритмов, вычислительные эксперименты.

2. Блочный алгоритм Гаусса-Зейделя в неявной форме для систем специального вида и с использованием красно-черного упорядочения

Красно-черное упорядочение сводит алгоритм зейделевского типа к двухфазной процедуре типа Якоби с сохранением точности и числа операций исходного алгоритма. Красно-черное упорядочение позволяет сохранить зейделевский тип вычислений при переходе к параллельной версии алгоритма. Без применения красно-черного упорядочения параллельные версии алгоритма несколько теряют точность вычислений, так как часть вычислений зейделевского типа заменяется вычислениями типа Якоби – особенность, возникающая при организации параллельных потоков вычислений.

Рассмотрим алгоритм, основанный на методе Гаусса-Зейделя в неявной форме. Если в этом алгоритме использовать красно-черное упорядочение, то получим

do $i = 0, N$
  $Y_i = Y_i^0 = \ldots$ // выбирается начальное приближение, $Y_0 = F_0 = 0$
enddo
do $l = 0, L - 1$ // L – заданное или предельное число итераций
  do $i = 2, N$, step=2
    $\Phi = F_i - A_i Y_{i-1}$ // используется Y_i с нечётным i
    if $i \neq N$ then $\Phi = \Phi - B_i Y_{i+1}$ // используется Y_i с нечётным i

Решить СЛАУ $C_i Y_i = \Phi$

enddo

do $i = 1, N$, step=2

$\Phi = F_i - A_i Y_{i-1}$ // используется Y_i с чётным i

if $i \neq N$ then $\Phi = \Phi - B_i Y_{i+1}$ // используется Y_i с чётным i

Решить СЛАУ $C_i Y_i = \Phi$

enddo

enddo(k)

На каждом шаге l требуются умножения блочно-двухдиагональных матриц (с вложенными блоками второго порядка) на вектор. Возникающие системы линейных алгебраических уравнений с блочно-трёхдиагональными матрицами (блоки второго порядка) можно решить методом матричной (матрицы второго порядка) прогонки.

3. Псевдокод параллельного алгоритма для реализации на многоядерных компьютерах с общей памятью

Пусть $Ax = f$ – СЛАУ порядка n. Для достижения критерия остановки итерационного процесса требуется на l-м итерационном шаге проанализировать норму вектора $r^{l+1} = f - Ax^{l+1}$ невязки системы: $\left\| r^{j+1} \right\|_\infty = \max\limits_{1 \le i \le n} \left| r_i^{j+1} \right|$ (используется максимум-норма). Потоки вычислений будем задавать циклом с параметром i.

Алгоритм. Блочный неявный метод Гаусса-Зейделя решения СЛАУ, ориентированной на гемодинамику.

Вход:

N, M – параметры СЛАУ;

ε – предельное значение нормы невязка, характеризует точность алгоритма;

двумерный массив, каждый элемент которого есть вектор размерности 2 – векторы начального приближения $Y_0^0, Y_1^0, ..., Y_N^0$:

$$y(i,k),\ 0 \le i \le N,\ 0 \le k \le M \ \ //\ \ y(0,k) = y_{0,k} = \begin{pmatrix} 0 \\ 0 \end{pmatrix},\ \ y(i,k) = y_{i,k} = \begin{pmatrix} y_{i,k}^u \\ y_{i,k}^w \end{pmatrix};$$

двумерные массивы, элементы которых есть матрицы порядка 2:

$$cC(i,k)=\begin{pmatrix} cC1u(i,k) & cC1w(i,k) \\ cC2u(i,k) & cC2w(i,k) \end{pmatrix},\ 1\le i\le N,\ 0\le k\le M$$

$$//\ cC(i,k)=c_k^{C_i}=\begin{pmatrix} c_{k,1}^{C_i,u} & c_{k,1}^{C_i,w} \\ c_{k,2}^{C_i,u} & c_{k,2}^{C_i,w} \end{pmatrix}$$

$$aC(i,k)=\begin{pmatrix} aC1u(i,k) & aC1w(i,k) \\ 0 & aC2w(i,k) \end{pmatrix},\ 1\le i\le N,\ 1\le k\le M$$

$$//\ aC(i,k)=a_j^{C_i}=\begin{pmatrix} a_{k,1}^{C_i,u} & a_{k,1}^{C_i,w} \\ 0 & a_{k,2}^{C_i,w} \end{pmatrix}$$

$$bC(i,k)=\begin{pmatrix} bC1u(i,k) & 0 \\ bC2u(i,k) & bC2w(i,k) \end{pmatrix},\ 1\le i\le N,\ 0\le k\le M-1$$

$$//\ bC(i,k)=b_k^{C_i}=\begin{pmatrix} b_{k,1}^{C_i,u} & 0 \\ b_{k,2}^{C_i,u} & b_{k,2}^{C_i,w} \end{pmatrix}$$

$$cA(i,k)=\begin{pmatrix} cA1u(i,k) & 0 \\ cA2u(i,k) & cA2w(i,k) \end{pmatrix},\ 1\le i\le N,\ 0\le k\le M$$

$$//\ cA(i,k)=c_k^{A_i}=\begin{pmatrix} c_{k,1}^{A_i,u} & 0 \\ c_{k,2}^{A_i,u} & c_{k,2}^{A_i,w} \end{pmatrix}$$

$$bA(i,k)=\begin{pmatrix} 0 & bA1w(i,k) \\ 0 & 0 \end{pmatrix},\ 1\le i\le N,\ 0\le k\le M-1$$

$$//\ bA(i,k)=b_k^{A_i}=\begin{pmatrix} 0 & b_{k,1}^{A_i,w} \\ 0 & 0 \end{pmatrix}$$

$$cB(i,k)=\begin{pmatrix} cB1u(i,k) & cB1w(i,k) \\ 0 & cB2w(i,k) \end{pmatrix},\ 1\le i\le N-1,\ 0\le k\le M$$

$$//\ cB(i,k)=c_k^{B_i}=\begin{pmatrix} c_{k,1}^{B_i,u} & c_{k,1}^{B_i,w} \\ 0 & c_{k,2}^{B_i,w} \end{pmatrix}$$

$$aB(i,k)=\begin{pmatrix} 0 & aB1w(i,k) \\ 0 & 0 \end{pmatrix},\ 1\le i\le N-1,\ 1\le k\le M$$

$$//\ aB(i,k)=a_k^{B_i}=\begin{pmatrix} 0 & a_{k,1}^{B_i,w} \\ 0 & 0 \end{pmatrix};$$

двумерный массив, элементы которого есть векторы размерности 2:

$$fslae(i,k)=\begin{pmatrix} fu(i,k) \\ fw(i,k) \end{pmatrix},\ 1\le i\le N,\ 0\le k\le M\ \ //\ \ fslae(0,k)=f_{0,k}=\begin{pmatrix} 0 \\ 0 \end{pmatrix};$$

подпрограмма умножения матрицы порядка 2 на двумерный вектор;

подпрограмма матричной прогонки для решения системы, все коэффициенты матрицы которой – матрицы-блоки порядка 2, а каждый элемент вектора неизвестных и вектора правой части – двумерный подвектор.

Выход:

двумерный массив, элементы которого есть векторы размерности 2:

$$y(i,k),\ 0\le i\le N,\ 0\le k\le M\ \ //\ \ y(i,k)=y_{i,k}=\begin{pmatrix} y^u_{i,k} \\ y^w_{i,k} \end{pmatrix}.$$

do $i = 0, N$

$Y_i = Y_i^0 = \ldots$ // выбирается начальное приближение, $Y_0 = F_0 = 0$

enddo

do $l = 0, 1, 2, \ldots$ (пока не достигнут критерий остановки $\left\| r^{j+1} \right\| < \varepsilon$):

// итерации следующего цикла распределяются между потоками

dopar $i = 2, N$, step=2

// начало получения вектора $\Phi = F_i - A_i Y_{i-1} - B_i Y_{i+1}$, если $i \ne N$ или вектора $\Phi = F_i - A_i Y_{i-1}$, если $i = N$; используется Y_i с нечётным i

do $k = 0, M-1$

$phi(k) = fslae(i,k) - cA(i,k)y(i-1,k) - bA(i,k)y(i-1,k+1)$

// φ_k – k-й элемент Φ; умножения – матрично-векторные

enddo(k)

$phi(M) = fslae(i,M) - cA(i,M)y(i-1,M)$

if $i \ne N$ do

$phi(0) = phi(0) - cB(i,0)y(i+1,0)$

do $k = 1, M$

$phi(k) = phi(k) - cB(i,k)y(i+1,k) - aB(i,k)y(i+1,k-1)$

enddo(k)

endif($i \ne N$)

// конец получения вектора $\Phi = F_i - A_i Y_{i-1} - B_i Y_{i+1}$, если $i \ne N$ или вектора $\Phi = F_i - A_i Y_{i-1}$, если $i = N$

Решение СЛАУ $C_i Y_i = \Phi$ методом матричной прогонки

enddopar($i = 2, N$, step=2)

// итерации следующего цикла распределяются между потоками
dopar $i = 1, N$, step=2
операторы тела цикла dopar $i = 2, N$, step=2
// используется Y_i с чётным i
enddopar($i = 1, N$, step=2)
enddo(l)

4. Вычислительные эксперименты

Для вычислительных экспериментов использовался компьютер с общей памятью с 12 физическими ядрами (логических потоков – 20). Вычисления проводились на модельном примере при $N = 1000$, $M = 3000$ (рис. 1) и $N = 3000$, $M = 3000$ (рис. 2), $\varepsilon = 10^{-6}$. Сравнивалось время выполнения четырёх алгоритмов. На рисунках синий цвет графика соответствует последовательному алгоритму (SeqGS), оранжевый – последовательному алгоритму с использованием красно-черного упорядочения (SeqRBGS),

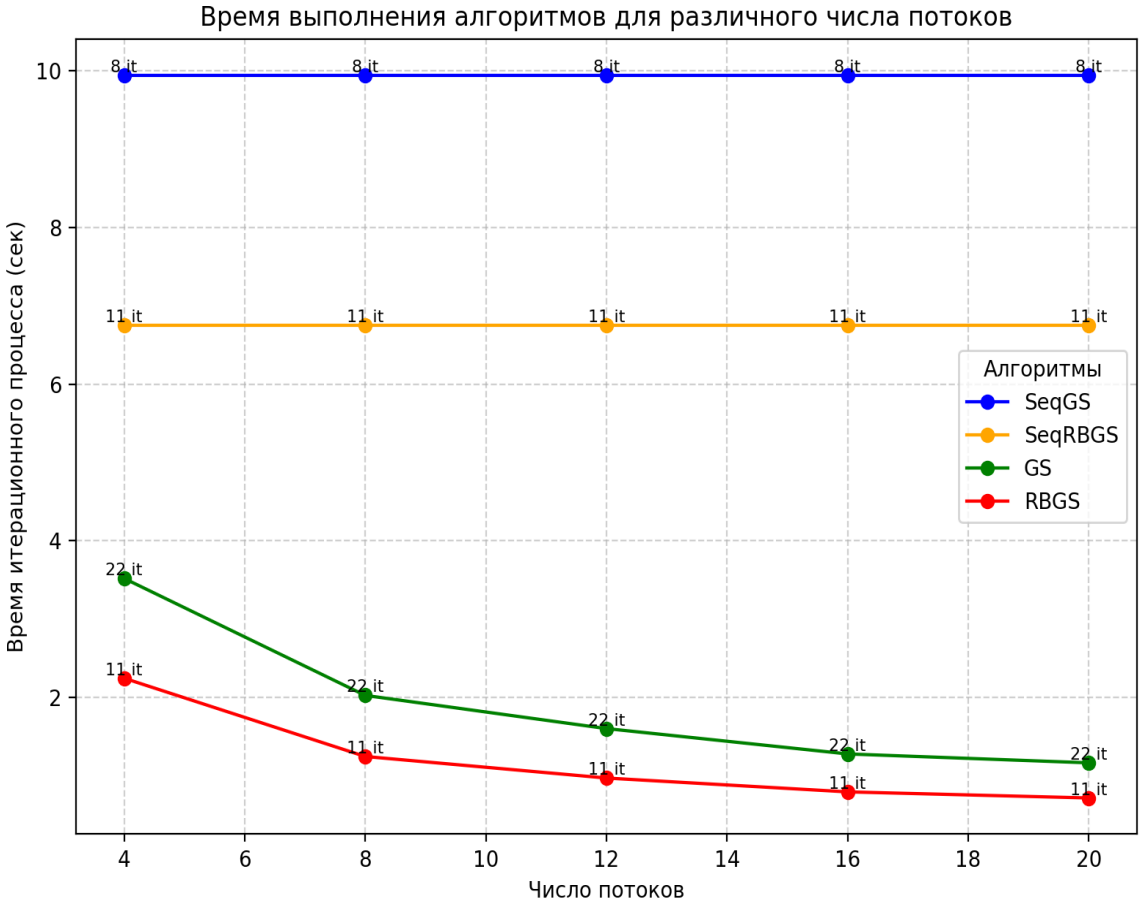

Рис. 1. Зависимость времени работы алгоритмов от количества потоков, $N = 1000$, $M = 3000$

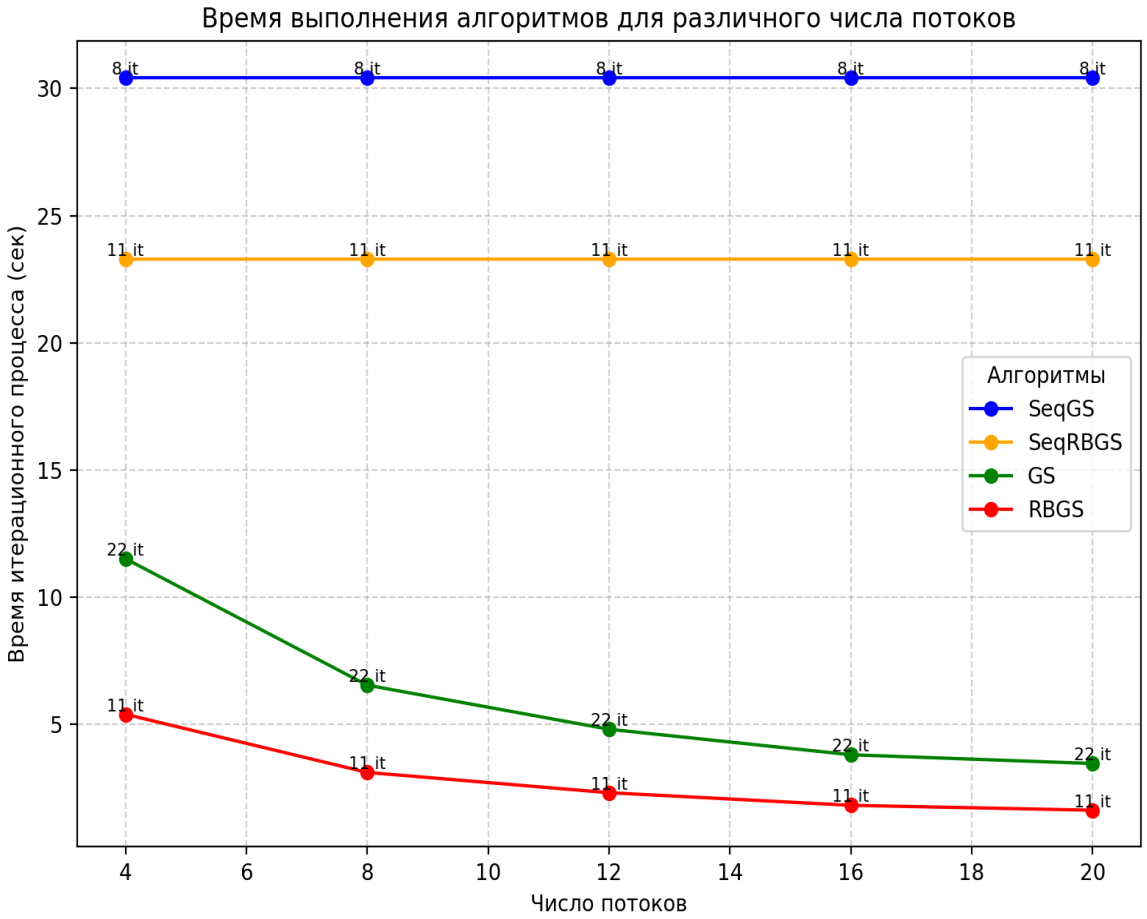

Рис. 2. Зависимость времени работы алгоритмов от количества потоков, $N = 3000$, $M = 3000$

зеленый – параллельному алгоритму (GS), красный – параллельному алгоритму с использованием красно-черного упорядочения (RBGS). Последовательные алгоритмы реализованы на языке программирования C++, параллельные – на языке C++ с использованием OpenMP. На графиках также указано количество итераций, необходимых для достижения требуемой точности.

Меньшее время выполнения алгоритма SeqRBGS, по сравнению с алгоритмом SeqGS, несмотря на большее количество выполненных итераций, можно объяснить лучшим использованием памяти с быстрым доступом. Из графиков видно, что наименьшее время для достижения требуемой точности достигается для блочного параллельного алгоритма Гаусса-Зейделя с использованием красно-черного упорядочения, он значительно превосходит другой параллельный алгоритм.