Лабораторная работа №1

«Параллельная реализация решения системы линейных алгебраических уравнений с помощью MPI»

РЕШЕНИЕ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ ИТЕРАЦИОННЫМИ МЕТОДАМИ

Пусть есть система из N линейных алгебраических уравнений в виде $Ax=b$, где A – матрица коэффициентов уравнений размером $N \times N$, b–вектор правых частей размером N, x–искомый вектор решений размером N. Решение системы уравнений итерационным методом состоит в выполнении следующих шагов.

1. Задается x^0 – произвольное начальное приближение решения (вектор с произвольными начальными значениями).
2. Приближение многократно улучшается с использованием формулы вида $x^{n+1} = f(x^n)$, где функция f определяется используемым методом[1].
3. Процесс продолжается, пока не выполнится условие $g(x^n) < \varepsilon$, где функция g определяется используемым методом, а величина ε задает требуемую точность.

Ниже представлено несколько итерационных методов решения систем линейных алгебраических уравнений.

Метод простой итерации

В методе простой итерации преобразование решения на каждом шаге задается формулой:

$$x^{n+1} = x^n - \tau(Ax^n - b).$$

Здесь τ – константа, параметр метода. В зависимости от значения параметра τ последовательность $\{x^n\}$ может сходиться к решению быстрее или медленнее, или вообще расходиться. В качестве подходящего значения τ

[1]Общая формула для итерационных методов выглядит следующим образом: $x^{n+1} = f(x^{n+1}, x^n, x^{n-1}, \ldots, x^0)$, но для целей лабораторных работ достаточно будет формулы, представленной в тексте.

можно взять 0.01 или -0.01. Знак параметра τ зависит от задачи. Если с некоторым знаком решение начинает расходиться, то следует сменить его на противоположный. Критерий завершения счета:

$$\frac{\|Ax^n - b\|_2}{\|b\|_2} < \varepsilon,$$

где $\|u\|_2 = \sqrt{\sum_{i=0}^{N-1} u_i^2}$. Для тестирования метода значение ε можно взять равным 10^{-5}.

Метод минимальных невязок

В методе минимальных невязок преобразование решения на каждом шаге задается следующими формулами:

$$y^n = Ax^n - b,$$

$$\tau^n = \frac{(y^n, Ay^n)}{(Ay^n, Ay^n)},$$

$$x^{n+1} = x^n - \tau^n y^n,$$

где $(u, v) = \sum_{i=0}^{N-1} u_i v_i$. В отличие от метода простой итерации, метод минимальных невязок содержит два умножения матрицы на вектор на каждой итерации, однако сходится до нужной точности за меньшее число итераций. Критерий завершения счета можно взять такой же, как в методе простой итерации.

Метод сопряженных градиентов

В методе сопряженных градиентов преобразование решения на каждом шаге задается следующими формулами:

$$r^0 = b - Ax^0,$$

$$z^0 = r^0,$$

$$\alpha^{n+1} = \frac{(r^n, r^n)}{(Az^n, z^n)},$$

$$x^{n+1} = x^n + \alpha^{n+1} z^n,$$
$$r^{n+1} = r^n - \alpha^{n+1} A z^n,$$
$$\beta^{n+1} = \frac{(r^{n+1}, r^{n+1})}{(r^n, r^n)},$$
$$z^{n+1} = r^{n+1} + \beta^{n+1} z^n.$$

где $(u, v) = \sum_{i=0}^{N-1} u_i v_i$. Метод сопряженных градиентов на каждой итерации содержит только одно умножение матрицы на вектор и, кроме того, сходится быстрее, чем предыдущие два метода. Однако метод сопряженных градиентов работает только для симметричных матриц коэффициентов. Критерий завершения счета в методе сопряженных градиентов следующий:

$$\frac{\|r^n\|_2}{\|b\|_2} < \varepsilon,$$

где $\|u\|_2 = \sqrt{\sum_{i=0}^{N-1} u_i^2}$.Для выполнения лабораторных работ значение εможно взять равным 10^{-5}.

ИСХОДНЫЕ ДАННЫЕ

Исходные данные для тестирования реализаций представленных методов и выполнения лабораторной работы можно взять следующие.

Модельная задача с заданным решением

Элементы главной диагонали матрицы *A* равны 2.0, остальные равны 1.0.Все элементы вектора *b* равны *N*+1. В этом случае решением системы будет вектор, элементы которого равны 1.0. Начальные значения элементов вектора *x* можно взять равными 0.

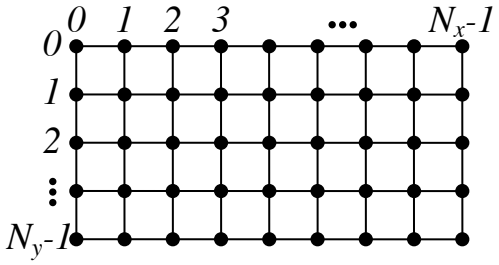
Модельная задача с произвольным решением

Элементы главной диагонали *A* равны 2.0, остальные равны 1.0. Формируется вектор *u*, элементы которого заполняются произвольными значениями, например $u_i = \sin \frac{2\pi i}{N}$. Элементы вектора *b* получаются путем умножения матрицы *A* на вектор *u*. В этом случае решением системы будет

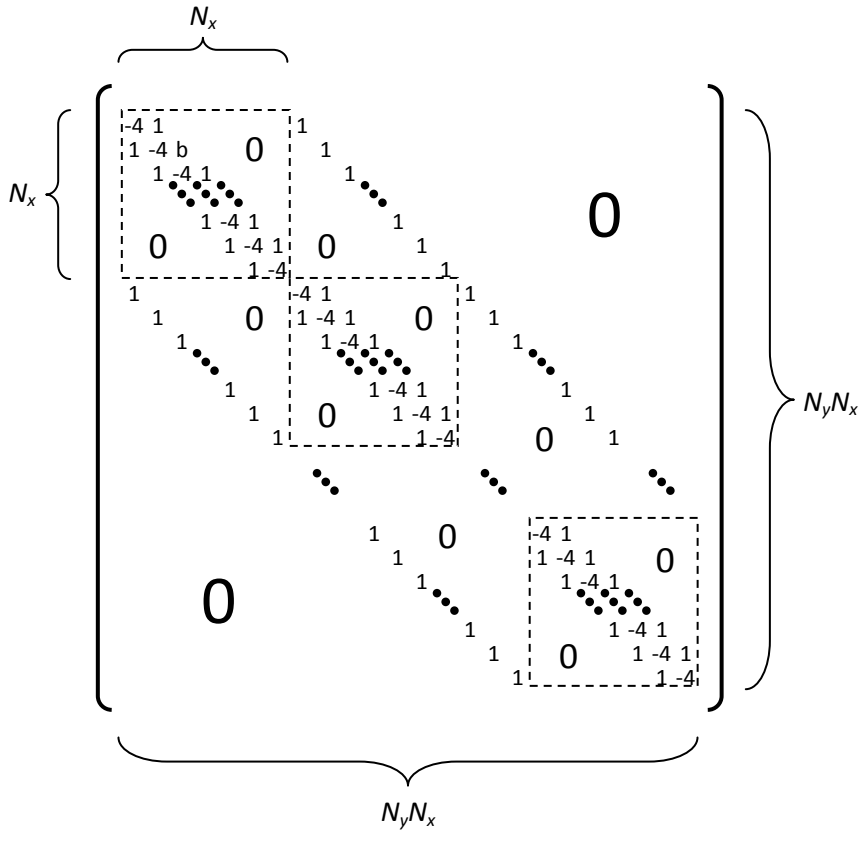
вектор, равный вектору *и*.Начальные значения элементов вектора *x* можно взять равными 0.

Задача о распределении тепла в пластинке

Есть прямоугольная металлическая пластинка, которую в некоторых точках нагревают, а в некоторых – охлаждают. Необходимо вычислить стационарное распределение температуры на пластинке. Для этого представим пластинку в виде прямоугольной сетки размера $N_y \times N_x$. Будем искать температуру в узлах этой сетки.



Определить распределение температуры на сетке можно путем решения системы линейных алгебраических уравнений со следующей матрицей размера $N \times N$, где $N = N_y N_x$.



Значения вектора правой части системы уравнений задают источники и стоки тепла в узлах сетки: значения меньше нуля задают источники тепла, значения больше нуля – стоки, нулевые значения – отсутствие источников и стоков. Узел (i,j) прямоугольной сетки, где $i = 0,1,...,N_y-1$ – координата узла по оси Y, а$j = 0,1,...,N_x-1$ – координата узла по оси X, соответствует элементу вектора правой части (или вектора решения) с номером $k = (i*N_x)+j$. Вектор правой части bможно целиком заполнить нулями кроме нескольких ненулевых значений в диапазоне [-50, 50] в произвольных местах вектора. Начальные значения элементов вектора xможно взять равными 0.

Полученное решение можно посмотреть наглядно с помощью программы gnuplot. Для этого необходимо вывести вектор решения x в бинарный файл, например «result.dat», и выполнить в том же каталоге команду:

```
gnuplot -e "plot 'result.dat' binary array=(50,30)\
format='%lf' with image ; pause -1"
```

Вместо чисел 50 и 30 следует указать свои размеры N_y и N_x.

ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ

1. Написать программу на языке C или C++, которая реализует итерационный алгоритм решения системы линейных алгебраических уравнений вида $Ax=b$ в соответствии с выбранным вариантом. Здесь A – матрица размером $N \times N$, x и b – векторы длины N. Тип элементов – double.
2. Программу распараллелить с помощью MPI с разрезанием матрицы A по строкам на близкие по размеру, возможно не одинаковые, части. Соседние строки матрицы должны располагаться в одном или в соседних MPI-процессах. Реализовать два варианта программы:
    - Вариант 1: векторы x и b дублируются в каждом MPI-процессе,
    - Вариант 2: векторы x и b разрезаются между MPI-процессами аналогично матрице A.

Уделить внимание тому, чтобы при запуске программы на различном числе MPI-процессов решалась одна и та же задача (исходные данные заполнялись одинаковым образом).

3. Замерить время работы двух вариантов программы при использовании различного числа процессорных ядер: 1,2, 4, 8, 16. Построить графики зависимости времени работы программы, ускорения и эффективности распараллеливания от числа используемых ядер. Исходные данные, параметры N и ε подобрать таким образом, чтобы решение задачи на одном ядре занимало не менее 30 секунд.
4. Выполнить профилирование двух вариантов программы с помощью MPE при использовании 16-и ядер.
5. На основании полученных результатов сделать вывод о целесообразности использования одного или второго варианта программы.

ВАРИАНТЫ ЗАДАНИЙ

1. Метод простой итерации.
2. Метод минимальных невязок.
3. Метод сопряженных градиентов.