

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Национальный исследовательский
Нижегородский государственный университет им. Н.И. Лобачевского»
(ННГУ)

Институт информационных технологий, математики и механики

Кафедра математического обеспечения и суперкомпьютерных технологий

Направление подготовки: «Прикладная математика и информатика»
Профиль подготовки: «Вычислительная математика и суперкомпьютерные
технологии»

Отчет по лабораторной работе №3

Выполнил: студент группы 381903-3м

_____ Панов А.А.
Подпись

Проверил:

к.ф.-м. н., доц., доцент каф. МОСТ

_____ Баркалов К.А.
Подпись

Нижний Новгород

2020

Содержание

Введение.....	3
1. Постановка задачи.....	4
2. Метод частичной дискретизации.....	5
3. Метод Рунге-Кутты 4 порядка.....	6
4. Параллельная реализация.....	7
5. Вычислительные эксперименты	8
6. Заключение	11
Список литературы	12

Введение

Многие физические задачи описываются дифференциальными уравнениями в частных производных. Очень часто аналитически решить такие уравнения невозможно. Такие уравнения обычно решаются численно. В данной лабораторной на примере решения динамической одномерной задачи теплопроводности (задача о «стержне») будут рассмотрены параллельные реализации некоторых методов.

1. Постановка задачи

С помощью метода частичной дискретизации, используя метод Рунге-Кутты 4-ого порядка решить динамическую одномерную задачу теплопроводности вида:

$$\frac{du(x, t)}{dt} = \frac{d^2u(x, t)}{dx^2} + f(x, t), x \in [0, L], t \in [0, T]$$

Со следующими граничными/начальными условиями:

$$\begin{aligned} u(0, t) &= v_0(t) \\ u(L, t) &= v_n(t) \\ u(x, 0) &= u_0(x) \end{aligned}$$

Требуется найти численное решение $u(x, T)$ на равномерной сетке:

$$G = \left\{ (x_i, t_j) : x_i = ih, t_j = j\tau, i = \overline{0, n}, j = \overline{0, m} \right\}$$

$$h = \frac{L}{n}, \tau = \frac{T}{m}$$

Программа на языке C++ должна реализовывать функцию со следующим заголовком:

```
class heat_task {
public:
    double T; // момент времени, в который необходимо аппроксимировать
    u(x, t)
    double L; // длина стержня
    int n; // размер сетки по x
    int m; // размер сетки по t
    double initial_condition(double x); // функция, задающая начальное
    условие
    double left_condition (double t); // функция, задающая граничное
    условие при x = 0
    double right_condition (double t); // функция, задающая граничное
    условие при x = L
    double f(double x, double t); // функция, задающая внешнее
    воздействие
};
void heat_equation_runge_kutta(heat_task task, double * v);
```

Должны быть выполнены условия на порядок сходимости: $O(\tau^4 + h^2)$

Размерность сетки $nm \leq 2^{26}$.

2. Метод частичной дискретизации

Рассмотрим простейшую версию уравнения теплопроводности:

$$\frac{du(x, t)}{dt} = \frac{d^2u(x, t)}{dx^2}, \quad x \in [0, 1], t \in [0, 1]$$

$$u(0, t) = 0$$

$$u(1, t) = 0$$

$$u(x, 0) = u_0(x)$$

Аппроксимируем вторую производную разностным оператором второго порядка:

$$\frac{u(x_i, t)}{dt} \approx \frac{u(x_{i+1}, t) - 2u(x_i, t) + u(x_{i-1}, t))}{h^2}$$

Выполним замену: $u(x_i, t) = v^i(t)$, после этого в матричном виде можно записать:

$$v'(t) = \frac{1}{h^2} Av(t), \text{ матрица } A \text{ трех диагональная, с элементами } 1, -2, 1$$

Далее к этому выражению нужно применить метод Рунге-Кутты.

3. Метод Рунге-Кутта 4 порядка

С помощью данного метода выражение вида $v'(t) = F(t, v) = \frac{1}{h^2} Av(t)$ можно вычислять итерационно:

$$v^{n+1} = v^n + \frac{\tau}{6} (k_1 + 2k_2 + 2k_3 + k_4)$$

$$k_1 = f(t^n, v^n)$$

$$k_2 = f(t^n + \frac{\tau}{2}, v^n + \frac{\tau}{2} k_1)$$

$$k_3 = f(t^n + \frac{\tau}{2}, v^n + \frac{\tau}{2} k_2)$$

$$k_4 = f(t^n + \tau, v^n + \tau k_3)$$

В данном случае, так как A матрица, то v^{n+1} и коэффициенты k_i будут векторными.

Чтобы данный метод сходил, необходимо чтобы шаг по времени был меньше квадрата шага по x , т. е. $\tau < h^2$.

Лабораторной поставлена немного другая задача (с ненулевыми граничными условиями, и с внешней функцией воздействия f), но полученные формулы легко дополняются, в частности функция F будет иметь вид:

$$F(t, v) = \frac{1}{h^2} Av(t) + f(t) + (v^0(t), 0, \dots, 0, v^n(t))^T$$

4. Параллельная реализация

Реализация кода по формулам выполняется несложно (хотя нужно соблюдать аккуратность с коэффициентами).

Компоненты каждого векторного коэффициента $k_1, k_2 \dots$ можно вычислять параллельно. Также компоненты v^{n+1} можно вычислять параллельно. Единственная проблема, что внутри таких параллельных циклов довольно мало вычислений. В таком случае доля накладных расходов связанных с синхронизацией после каждого параллельного цикла, довольно большая, что плохо сказывается на масштабируемости. Также «вредит» большое количество операций чтения/записей относительно количества вычислительных операций.

Общий вычислительный цикл выглядит так:

```
for (int tick = 0; tick < m; tick++)
{
    curTime = dt * tick;
    v[0] = task.left_condition(curTime);
    v[n] = task.right_condition(curTime);
    calc_k1(task, v, k1, curTime);
    calc_k2(task, v, k1, k2, curTime + dt/2, tmp.data());
    calc_k3(task, v, k2, k3, curTime + dt/2, tmp.data());
    calc_k4(task, v, k3, k4, curTime + dt, tmp.data());
    calc(task, v, k1, k2, k3, k4, curTime);
}
```

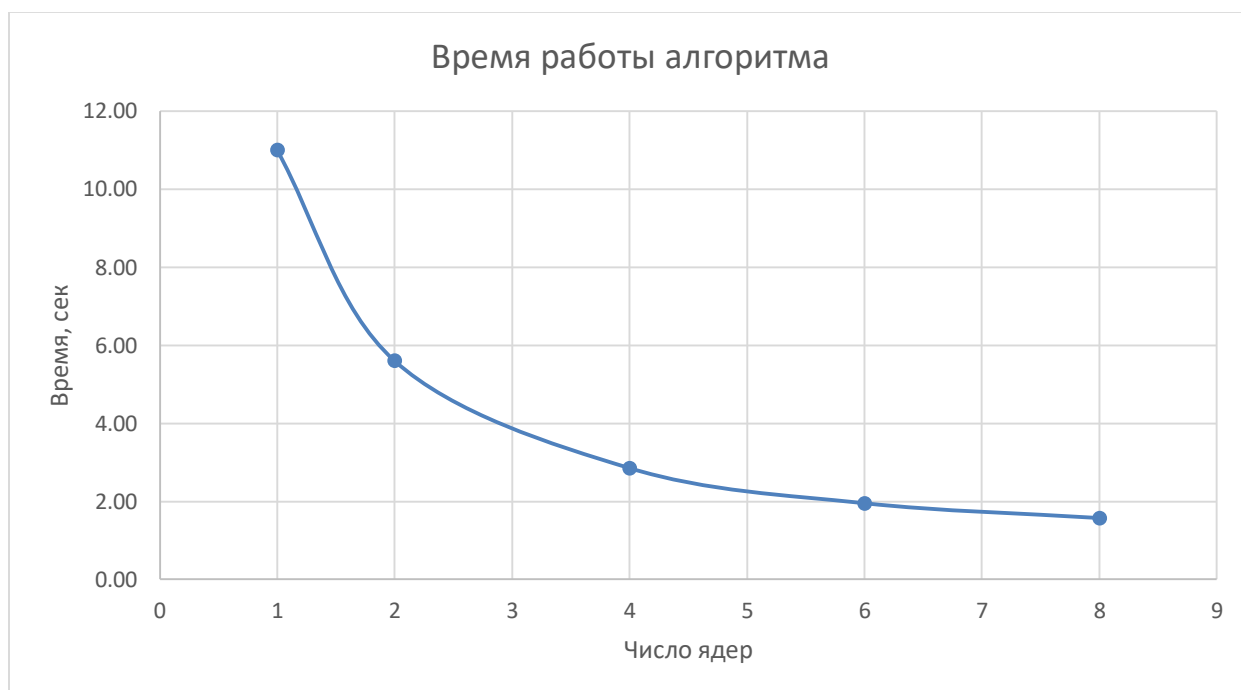
5. Вычислительные эксперименты

Решалась задача с длиной стрижня $L=100$, временем $T = 0.1$, $n = 50000$, $m = 50000$ (шаг по времени должен быть ограничен шагом по пространству),

Запуск производился на восьми ядерном процессоре Intel core i7 9700K, 16 gb ОЗУ. Использовался Intel® C++ Compiler 19.1 for Windows* с оптимизацией O2.

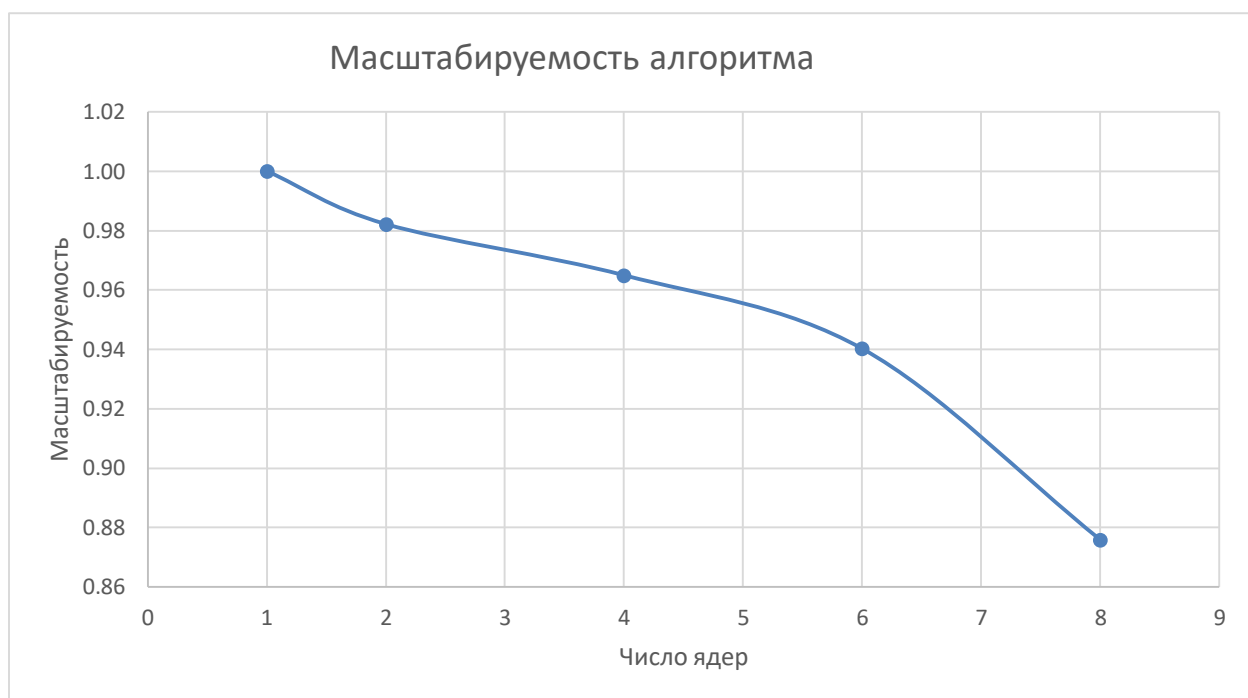
Время, сек.	1 core	2 core	4 core	6 core	8 core
	11.00	5.60	2.85	1.95	1.57

Таблица 1. Время решения задачи.



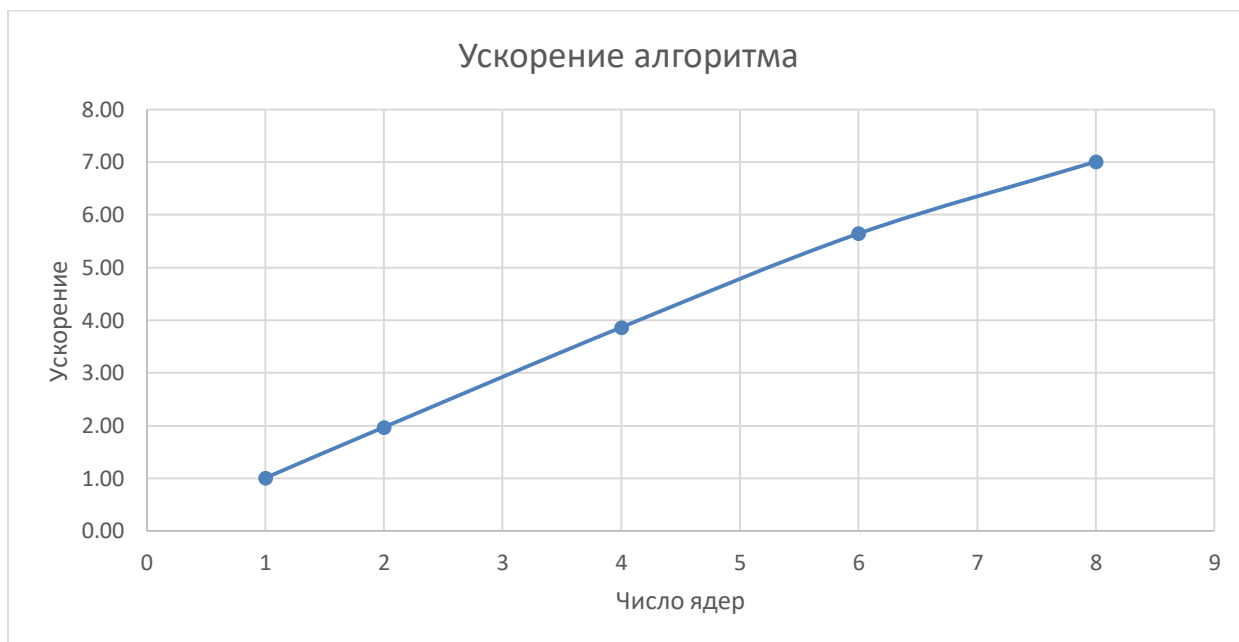
Масштабируемость	1 core	2 core	4 core	6 core	8 core
	1.00	0.98	0.96	0.94	0.88

Таблица 2. Масштабируемость алгоритма.



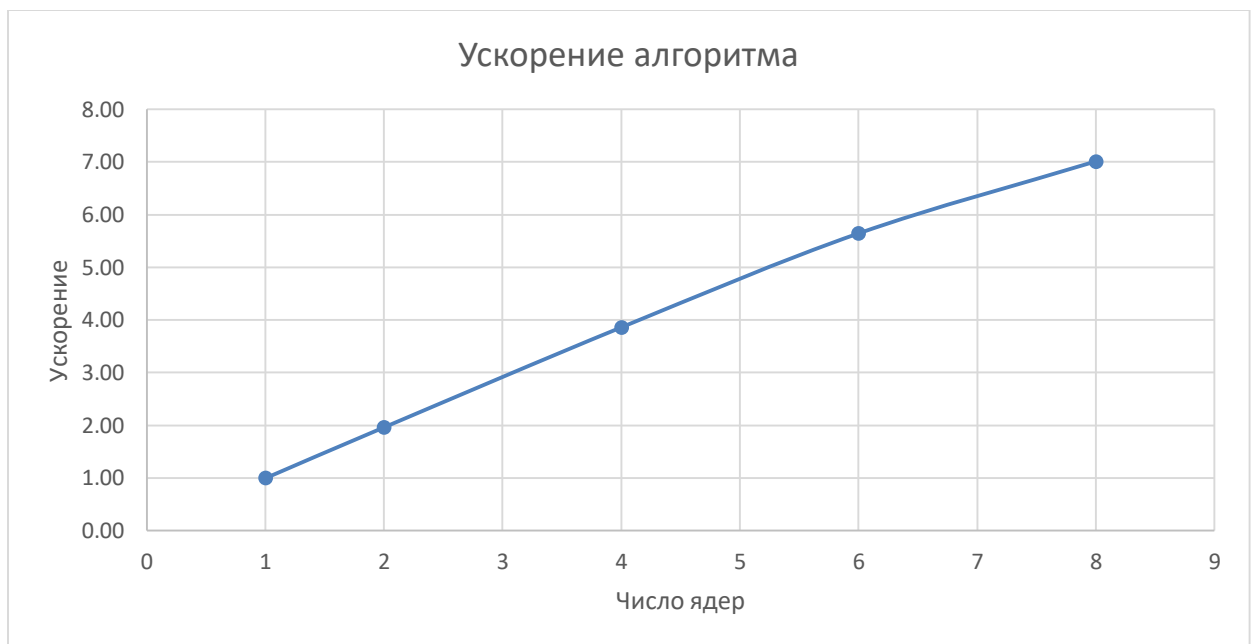
Ускорение	1 core	2 core	4 core	6 core	8 core
	1.00	1.96	3.86	5.64	7.01

Таблица 3. Ускорение алгоритма.



6. Заключение

Численный эксперимент показал, что можно получить довольно неплохую масштабируемость (88% на 8 ядрах) при правильно подобранных n и m . Важно, чтобы n было довольно большим, чтобы время синхронизации занимало малую долю времени от общего времени вычислений. Еще одна проблема заключается в том, что параллелится цикл, сложность которого линейно зависит от количества данных. Таким образом увеличение размера данных, увеличивает число чтений/записей пропорционально числу вычислений. Таким образом, производительность упирается в скорость «памяти», а не процессора. Можно заметить, что чем сложнее функция f , тем больше будет вычислений, тем лучшей масштабируемости можно будет добиться. В целом, распараллеливание методов частичной дискретизации и Рунге-Кутты может принести значительное ускорение (в 7 раз на 8 ядрах).



Список литературы

1. Баркалов К.А. Параллельные численные методы.
2. Самарский А. А. Введение в численные методы. – Лань, 2009.