МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ 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**

Направление подготовки: «Фундаментальная информатика и информационные

технологии»

Магистерская программа: «Инженерия программного обеспечения»

**ОТЧЕТ**

по лабораторной работе

#### **«Волновая схема на задаче Дирихле для уравнения Пуассона»**

**Выполнил:** студент группы382006-1м

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.А. Солуянов

**Проверил:**

к.ф.-м. н., доц., доцент каф. МОСТ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_К.А. Баркалов

Нижний Новгород

2021

Оглавление

[Введение 3](#_Toc73047384)

[Постановка задачи 4](#_Toc73047385)

[Погрешность аппроксимации 5](#_Toc73047386)

[Метод верхних релаксаций 5](#_Toc73047387)

[Реализация 7](#_Toc73047388)

[Описание алгоритма 7](#_Toc73047389)

[Схема распараллеливания 8](#_Toc73047390)

[Результаты 9](#_Toc73047391)

[Тестовая инфраструктура 9](#_Toc73047392)

[Эксперименты 9](#_Toc73047393)

[Заключение 11](#_Toc73047394)

[Литература 12](#_Toc73047395)

# Введение

Математическое моделирование задач механики, физики и других отраслей науки и техники сводятся к дифференциальным уравнениям. В связи с этим решение дифференциальных уравнений является одной из важнейших математических задач. В вычислительной математике изучаются численные методы решения дифференциальных уравнений, которые особенно эффективны в сочетании с использованием вычислительной техники.

Задача Дирихле для уравнения Пуассона является одной из классических задач математической физики. Для решения уравнений с частными производными как правило используются сеточные методы, Нередко с помощью компьютера в области определения строится сетка, и составляется разностное уравнение, в котором искомыми неизвестными являются значения функции в узлах сетки. Решение разностного уравнение также можно искать по-разному. На практике широко применяются итерационные методы. Вычислительная схема в этом случае описывает, как следующее состояние сетки зависит от предыдущего. В результате счета на компьютере получается приближенное решение уравнений с частными производными.

Бывают сложные модели, в которых строятся сетки с большим количеством узлов: десятки миллионов и даже больше. Актуальной задачей для таких моделей является распараллеливание вычислений с целью сокращения времени счета.

В данной лабораторной работе будет рассмотрен метод релаксации с оптимальным параметром для задачи Дирихле для уравнения Пуассона с волновой схемой параллелизации вычислений.

# Постановка задачи

Рассмотрим уравнение Пуассона

|  |  |
| --- | --- |
|  | (1) |

Будем искать его решение, непрерывное в прямоугольнике

|  |  |
| --- | --- |
|  |  |

и принимающее на границе заданные значения

|  |  |
| --- | --- |
|  | (2) |

Задача, определяемая уравнением (1) и условием (2), называется задачей Дирихле.

Физическая интерпретация задачи – изгиб упругой пластины.

Определим разностную схему для задачи (1), (2). Введем в сетку и обозначим через сеточную функцию заданную на – шаги сетки по координатами .

Чтобы написать разностную схему для (1), (2), аппроксимируем каждую из производных на трехточечным шаблоне, полагая

|  |  |
| --- | --- |
|  | (3) |

|  |  |
| --- | --- |
|  | (4) |

Пользуясь выражениями (3), (4), заменим (1) разностным уравнением

|  |  |
| --- | --- |
|  | (5) |

К этому уравнению надо присоединить краевые условия

|  |  |
| --- | --- |
|  | (6) |

Уравнения (5) и (6) в совокупности образуют систему линейных алгебраических уравнений (СЛАУ), где число уравнений системы равно , т.е. столько, сколько и неизвестных, это значения при .

## Погрешность аппроксимации

Пусть – решение задачи Дирихле (1), (2), а решение разностной задачи (5), (6). Рассмотрим погрешность

Подставляя В (5) и (6), получаем для погрешности неоднородное уравнение

|  |  |
| --- | --- |
|  | (7) |

с однородным краевым условием

|  |  |
| --- | --- |
|  | (8) |

Здесь

|  |  |
| --- | --- |
|  | (9) |

Есть невязка или погрешность аппроксимации для схемы (5) на решении .

Можно показать, что

|  |  |
| --- | --- |
|  | (10) |

где

|  |  |
| --- | --- |
|  | (11) |

А значит схема (5) имеет второй порядок аппроксимации.

## Метод верхних релаксаций

Среди явных одношаговых итерационных методов наибольшее распространение получил метод верхних релаксаций. Это связано с тем, что метод верхних релаксаций содержит свободный параметр , изменяя который можно получать различную скорость сходимости итерационного процесса.

Наиболее эффективно этот метод применяется при решении множества близких алгебраических систем линейных уравнений. На первом этапе проводится решение одной из систем с различными значениями итерационного параметра и из анализа скорости сходимости итерационного процесса выбирается оптимальное значение этого параметра. Затем все остальные системы решаются с выбранным значением .

Еще одно достоинство итерационного метода верхних релаксаций состоит в том, что при его реализации на ЭВМ алгоритм вычислений имеет простой вид и позволяет использовать всего один массив для неизвестного вектора.

Основная вычислительная формула имеет вид

|  |  |
| --- | --- |
|  | (12) |

Метод релаксации для разностной схемы имеет вид

|  |  |
| --- | --- |
|  | (13) |

где

Оптимальный параметр метода релаксации:

|  |  |
| --- | --- |
|  | (14) |

# Реализация

## Описание алгоритма

Входными параметрами алгоритма является объект типа *heat\_task,* который содержит информацию о размерности сетки, размерах пластины и функции, определяющие граничные условия и внешнее воздействие.

Формат выхода должен выглядеть как указатель на массив, в котором содержится решение задачи.

Предварительными действиями является подготовка выходного вектора и матрицы внешнего воздействия, так как она неизменна и часто используется в алгоритме, поэтому выгодно посчитать ее заранее.

Таким образом основное тело алгоритма (Листинг 1) можно представить, как итеративное вычисление элементов результирующего вектора с помощью метода верхней релаксации.

Листинг 1. Основное тело алгоритма

for (int iteration = 0; iteration < max\_iter; iteration++) {

for (int k = 0; k < task.n + task.m - 3; ++k) {

int start = min(1 + k, task.n - 1);

int finish = max(1, k - task.m + 3);

#pragma omp parallel for private(prev, curr\_index, top\_index, bot\_index, left\_index, right\_index)

for (int i = start; i >= finish; --i) {

int j = task.m - (k - i + 2);

curr\_index = i \* (task.m + 1) + j;

top\_index = curr\_index + 1;

bot\_index = curr\_index - 1;

left\_index = curr\_index - (task.m + 1);

right\_index = curr\_index + (task.m + 1);

prev = v[curr\_index];

v[curr\_index] = omega \* (h\_ \* (v[left\_index] + v[right\_index]) + k\_ \* (v[top\_index] + v[bot\_index]) + f[i][j]) + \

(1 - omega) \* D \* prev;

v[curr\_index] /= D;

}

}

}

## Схема распараллеливания

Рассмотрим возможность построения параллельного алгоритма, который выполнял бы только те вычислительные действия, что и последовательный метод (может быть только в несколько ином порядке) и, как результат, обеспечивал бы получение точно таких же решений исходной вычислительной задачи. Как уже было отмечено выше, в последовательном алгоритме каждое очередное  приближение значения  вычисляется по последнему  приближению значений  и  и предпоследнему  приближению значений  и .

Таким образом, при требовании совпадения результатов вычислений последовательных и параллельных вычислительных схем в начале каждой итерации метода только одно значение  может быть пересчитано (возможности для распараллеливания нет). Но далее после пересчета  вычисления могут выполняться уже в двух узлах сетки  и  (в этих узлах выполняются условия последовательной схемы), затем после пересчета узлов  и  — в узлах ,  и  и т.д. Обобщая сказанное, можно увидеть, что выполнение итерации метода сеток можно разбить на последовательность шагов, на каждом из которых к вычислениям окажутся подготовленными узлы вспомогательной диагонали сетки с номером, определяемым номером этапа (Рисунок 1).

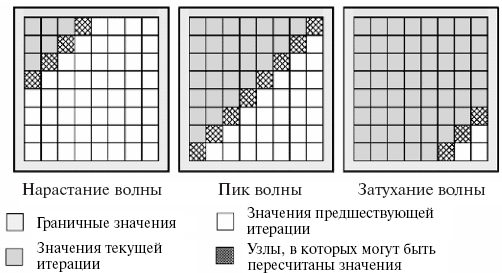


Рисунок 1. Волновая схема обработки данных

Такой алгоритм получил название волновой схемы. Следует отметить, что в нашем случае размер волны (степень возможного параллелизма) динамически изменяется в ходе вычислений – волна нарастает до своего пика, а затем затухает при приближении к правому нижнему узлу сетки.

# Результаты

## Тестовая инфраструктура

Вычислительные эксперименты проводились с использованием следующей инфраструктуры (Таблица 1).

|  |  |
| --- | --- |
| Процессор | 4 ядра, Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz 1.99 GHz |
| Память (RAM, Cache) | 8,00 ГБ, L1 – 256 Kb, L2 – 1 Mb, L3 – 8 Mb |
| Операционная система | Windows 10 |
| Среда разработки | Visual Studio 2019 |
| Компилятор | Intel(R) oneAPI DPC++/C++ Compiler 2021.1 |
| Библиотеки | OpenMP |

Таблица 1. Тестовая инфраструктура

## Эксперименты

Испытания проводились на задаче со следующими данными (Рисунок 2,3):

* Размер пластины
* Разбиения на сетку с числом узлов ,
* Граничные условия
  + Левая граница:
  + Левая граница:
  + Левая граница:
  + Левая граница:
* Функция внешнего воздействия

Рисунок 2. Зависимость времени работы программы от числа потоков

Рисунок 3. Зависимость ускорения от числа потоков

# Заключение

В ходе данной лабораторной работы был изучен метод релаксации с оптимальным параметром для задачи Дирихле для уравнения Пуассона с волновой схемой параллелизации вычислений.

Использование волновой схема обработки данных удалость достигнуть существенного ускорения алгоритма и избежать гонки данных, которая была бы неизбежна при попытке параллелизации последовательного обхода элементов матрицы.

# Литература

1. Самарский А.А. Введение численные методы. – СПб.: Лань, 2005
2. Самарский А.А., Гулин А.В. Численные методы. – М.: Наука, 1989.
3. Е.А. Рындин, И.В. Куликова, И.Е. Лысенко. Основы численных методов: теория и практика, 2015
4. Бахвалов Н.С., Жидков Н.П., Кобельков Г.М. Численные методы. – М.: Наука, 1987
5. Тихонов А.Н., Самарский А.А. Уравнения математической физики. – М.: Наука, 1977.