Министерство образования и науки Российской Федерации Федеральное агентство по образованию

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Вятский государственный университет»

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Исследование динамических топологий

Отчет по лабораторной работе №8 дисциплины

«Высокопроизводительные вычислительные комплексы»

Выполнил студент группы ИВТ-41\_\_\_\_\_\_\_\_\_\_/Жеребцов К. А./ Проверил /Мельцов В. Ю./

Киров 2023

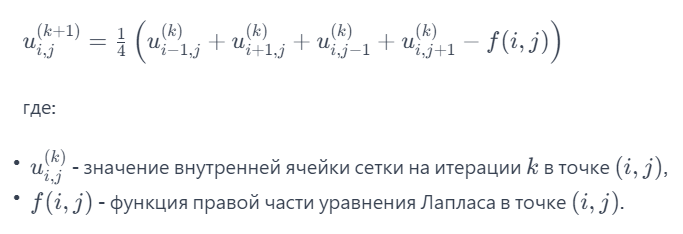
1. Цель работы

Получение практических навыков выполнения параллельных программ на кластере.

1. Ход работы
2. Задание

В качестве вычислительной задачи для распараллеливания был выбран итерационный метод Якоби для решения двумерного уравнения Лапласа.

1. Словесное описание алгоритма

Алгоритм Якоби для решения уравнения Лапласа в данном контексте основывается на итерационном обновлении значений внутри двумерной сетки с целью достижения стационарного состояния. Он применяется для численного решения уравнения Лапласа, которое в дискретной форме имеет вид:

1. **Инициализация:**
   * Создание двумерной сетки размера *N*×*N*.
   * Установка начальных значений внутренних ячеек сетки.
   * Установка граничных условий.
2. **Итерации:**
   * Повторение следующих шагов заданное количество раз (например, 5000 итераций):
     + Для каждой внутренней ячейки сетки применяется итерационная формула Якоби, обновляя ее значение на основе значений соседних ячеек и текущего приближения.
3. **Сходимость:**
   * Алгоритм повторяется до тех пор, пока изменения внутри сетки остаются небольшими или до достижения максимального числа итераций.
4. **Решение:**
   * После завершения итераций, значения внутренних ячеек сетки приближенно соответствуют решению уравнения Лапласа в заданных граничных условиях.

Для оценки степени параллелизма можно использовать формулу для ускорения:



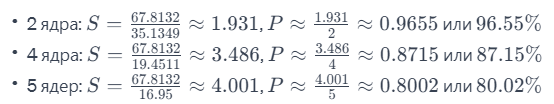
где *T*sequential​ - время выполнения на одном ядре, *T*parallel​ - время выполнения на *N* ядрах.

Степень параллелизма (*P*) может быть выражена как:

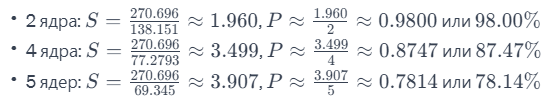


где *N* - количество ядер.

Для размера сетки 1000x1000 и 5000 итераций:



Для размера сетки 2000x2000 и 5000 итераций:



Оценка трудоемкости алгоритма Θ(n2) => чтобы повысить трудоемкость в m раз нужно увеличить n в

1. Результаты выполнения

В ходе выполнения лабораторной работы, написанная программа запускалась на различном числе ядер и для различных размерностей матриц.

Результаты экспериментов представлены в таблице 1.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Количество ядер, шт | Размер исходных матриц | | | | | | | | |
| 1000 | 1414 | 2000 | 2236 | 2828 | 3000 | 3464 | 3605 | 4000 |
| 1 | 19,19 | 38,88 | 78,76 | 98,96 | 160,16 | 181,03 | 244,31 | 265,48 | 330,37 |
| 2 | 9,70 | 19,75 | 39,98 | 50,32 | 81,68 | 92,36 | 124,96 | 135,77 | 169,21 |
| 4 | 5,20 | 10,51 | 21,55 | 27,17 | 44,62 | 50,65 | 69,03 | 75,42 | 95,00 |
| 5 | 5,32 | 9,66 | 18,67 | 23,19 | 37,28 | 41,90 | 56,89 | 62,24 | 78,05 |
| 8 | 3,64 | 6,67 | 12,61 | 15,52 | 25,00 | 28,26 | 38,44 | 41,94 | 53,27 |
| 9 | 3,77 | 6,36 | 11,26 | 14,02 | 22,69 | 25,46 | 34,66 | 38,30 | 48,00 |
| 12 | 3,47 | 5,23 | 9,06 | 11,36 | 17,78 | 20,15 | 27,42 | 29,90 | 37,63 |
| 13 | 3,32 | 4,89 | 8,88 | 10,97 | 16,87 | 19,08 | 25,82 | 28,16 | 35,58 |
| 16 | 2,99 | 4,50 | 7,67 | 9,37 | 14,54 | 16,45 | 22,64 | 24,10 | 30,82 |

На рисунках 1-3 представлены графики зависимостей:

* рисунок 1 – график зависимости времени выполнения от количества ядер;
* рисунок 2 – график зависимости времени выполнения от объема данных;
* рисунок 3 – график зависимости времени выполнения при равномерном увеличении числа ядер и объема данных.

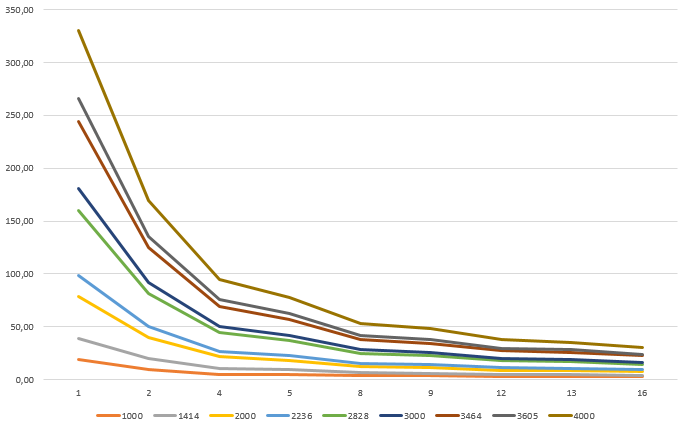


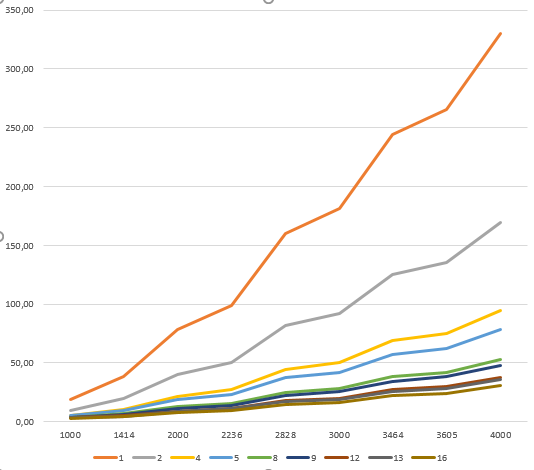
Рисунок 1 – Зависимость времени вычисления от числа ядер

Рисунок 2 – Зависимость времени вычисления от объема данных

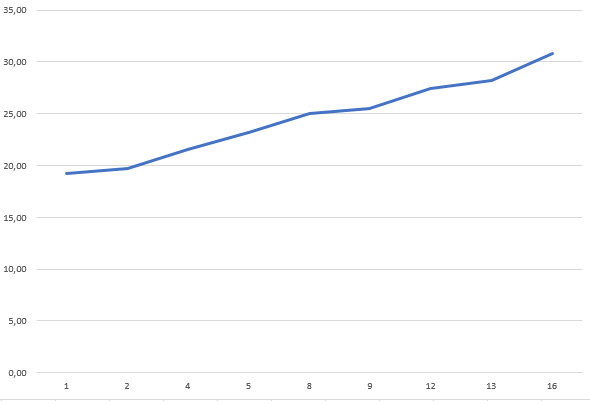


Рисунок 3 – Зависимость времени при одновременном увеличении числа ядер и объема данных

Вывод

При увеличении числа ядер и одинаковом объеме данных время решения задачи уменьшается. Наиболее значительное ускорения произошло при увеличении количества ядер с 1 до 4, при количестве ядер больше 4 эффективность параллельных вычислений снижается из-за накладных расходов на обмен данными.

При увеличении количества данных и неизменном количестве ядер время решения увеличивается сильнее при малом количестве ядер и слабо при большом количестве ядер. Разница в интенсивности изменения времени связана с тем, что при малом количестве ядер время расчёта в большей степени состоит из времени, затрачиваемом на решение задачи и в меньшей – на пересылку данных, поэтому оно так сильно зависит от размерности данных. В случае же большого количества ядер ситуация противоположная, и время меньше зависит от размерности данных, нежели от времени, затрачиваемого на их пересылку.

При пропорциональном увеличении количества ядер и объема исходных данных время решения задачи на практике увеличивается. Это можно объяснить тем, что время, затрачиваемое на пересылку данных между ядрами одного блейда значительно меньше, чем время, которое затрачивается на пересылку данных между блейдами.