Московский Государственный Университет им. М.В. Ломоносова
Факультет Вычислительной Математики и Кибернетики
Кафедра Суперкомпьютеров и Квантовой Информатики

Спецкурс: системы и средства параллельного программирования

Отчёт № 3

Исследование времени работы параллельного алгоритма «Блочное решето Эратосфена»

Работу выполнила
Домрачева Д. А.

Москва 2018

Постановка задачи и формат данных

Задача:

Реализовать параллельный алгоритм поиска простых чисел «решето Эратосфена» с помощью технологии MPI и оценить время работы алгоритма в зависимости от количества работающих одновременно процессов.

Формат командной строки:

<нижняя граница A> <верхняя граница B> <имя выходного файла>

Параметры командной строки, определяющие режим выполнения:

Программа находит все простые числа и их общее количество в диапазоне $[A;B]$ и выводит найденные простые числа в файл, указанные в командной строке.

Описание алгоритма

Математическая постановка:

Дан диапазон чисел $[A;B]$, необходимо найти все простые числа в этом диапазоне и их общее количество. Для этого необходимо:

1. Выписать подряд все числа от 2 до B.
2. Пусть переменная i изначально равна 2 – первое простое число.
3. Зачеркнуть в списке все числа от $2i$ до B с шагом $i(2i,3i,4i,\ldots)$.
4. Найти первое незачеркнутое число в списке, большее чем i, присвоить значение переменной i это число.
5. Повторять шаги 3 и 4, пока возможно.

Все оставшиеся незачеркнутые числа в диапазоне от A до B будут искомыми простыми числами.

Ресурс параллелизма:

Алгоритм можно сделать параллельным, вычислив и сохранив простые числа, не превосходящие $\sqrt{B}$, чтобы с их помощью вычеркивать числа, находящиеся в диапазоне $[max(A,\sqrt{B}+1);B]$, разделив этот отрезок между процессами.

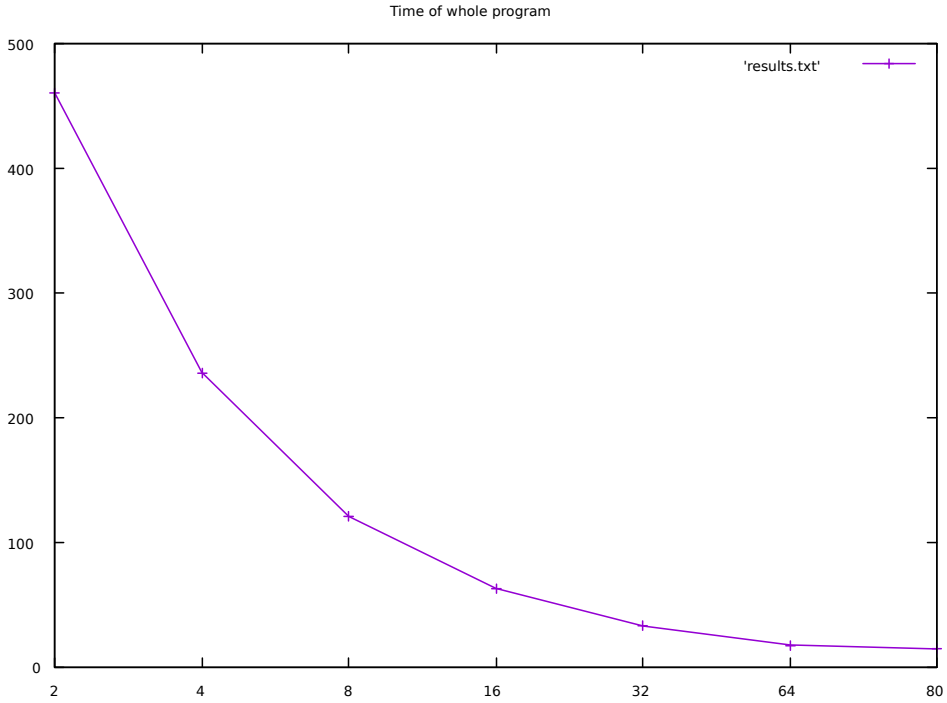
Верификация:

Для анализа правильности вывода программы использовался интернет-ресурс http://www.ega-math.narod.ru/Liv/Zagier.htm , где можно найти значения для количества простых чисел в некоторых тестовых диапазонах, а так же последовательная программа. Так, например, количество простых чисел в диапазоне $[1;100\,000\,000]-5\,761\,455$.

Результаты выполнения

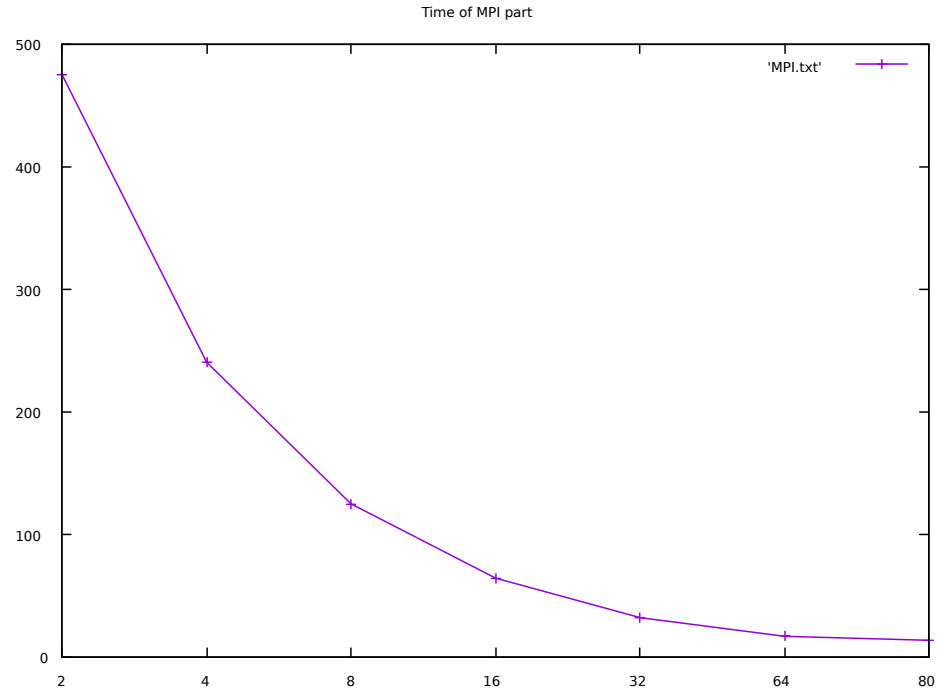
Общее время работы алгоритма (с учетом вывода и вычисления вспомогательного массива):

	Количество процессов						
	2	4	8	16	32	64	80
Время	476.71	241.54	126.07	65.31	33.34	17.69	14.06



Время работы MPI программы (без вывода и подсчета вспомогательного массива):

	Количество процессов						
	2	4	8	16	32	64	80
Время	475.31	240.13	124.66	63.77	31.64	16.27	12.99



Основные выводы

С увеличением количество процессов наблюдается экспоненциальное уменьшение времени работы программы.