Университет ИТМО

Факультет ПИиКТ

Высокопроизводительные системы

Лабораторная работа №1

Вариант 39

Выполнила: Соколова А.Д

Группа P33022

Санкт-Петербург

2021 г.

**Задание**

Реализовать алгоритмы сортировки в соответствии с вариантом задания. Параллелизовать реализованные алгоритмы с помощью OpenMP, применяя для каждого общий набор из разных подходов для распределения частей работы и объединения результатов. Собрать статистику о времени работы реализованных алгоритмов в целом и отдельных этапов в реализациях для различных наборов входных данныхв сочетании с различными степенями параллелизма. Сравнить все полученные реализации: разные алгоритмы попарно для каждого подхода и отдельно все реализации каждого алгоритма между собой. Объяснить результаты.

## **Вариант работы:**

Алгоритмы: Сортировка слиянием, пирамидальная

# **Выполнение работы**

## **Измерение времени для сортировки слиянием MergeSort**

LOG FOR 1 THREAD

size: 100000, time: 0.034000

size: 200000, time: 0.065000

size: 300000, time: 0.097000

size: 400000, time: 0.129000

size: 500000, time: 0.163000

size: 600000, time: 0.206000

size: 700000, time: 0.230000

size: 800000, time: 0.256000

size: 900000, time: 0.293000

size: 1000000, time: 0.315000

size: 5000000, time: 1.782000

size: 15000000, time: 5.414000

size: 25000000, time: 9.009000

size: 35000000, time: 12.660000

size: 45000000, time: 16.684000

size: 55000000, time: 20.215000

size: 65000000, time: 23.931000

size: 75000000, time: 28.268000

size: 85000000, time: 31.742000

size: 95000000, time: 36.062000

....................................................

LOG FOR 2 THREADS USING SECTIONS

size: 100000, time: 0.033000

size: 200000, time: 0.044000

size: 300000, time: 0.053000

size: 400000, time: 0.078000

size: 500000, time: 0.089000

size: 600000, time: 0.107000

size: 700000, time: 0.125000

size: 800000, time: 0.149000

size: 900000, time: 0.169000

size: 1000000, time: 0.172000

size: 5000000, time: 0.977000

size: 15000000, time: 2.982000

size: 25000000, time: 5.148000

size: 35000000, time: 7.473000

size: 45000000, time: 9.951000

size: 55000000, time: 12.823000

size: 65000000, time: 14.913000

size: 75000000, time: 17.784000

size: 85000000, time: 20.729000

size: 95000000, time: 22.724000

....................................................

LOG FOR 2 THREADS USING FOR

size: 100000, time: 0.027000

size: 200000, time: 0.048000

size: 300000, time: 0.056000

size: 400000, time: 0.075000

size: 500000, time: 0.095000

size: 600000, time: 0.113000

size: 700000, time: 0.134000

size: 800000, time: 0.154000

size: 900000, time: 0.162000

size: 1000000, time: 0.193000

size: 5000000, time: 0.848000

size: 15000000, time: 2.487000

size: 25000000, time: 4.744000

size: 35000000, time: 6.399000

size: 45000000, time: 8.796000

size: 55000000, time: 10.132000

size: 65000000, time: 12.002000

size: 75000000, time: 13.199000

size: 85000000, time: 14.623000

size: 95000000, time: 17.090000

....................................................

LOG FOR 4 THREADS USING FOR

size: 100000, time: 0.014000

size: 200000, time: 0.030000

size: 300000, time: 0.034000

size: 400000, time: 0.056000

size: 500000, time: 0.067000

size: 600000, time: 0.085000

size: 700000, time: 0.116000

size: 800000, time: 0.096000

size: 900000, time: 0.132000

size: 1000000, time: 0.144000

size: 5000000, time: 0.557000

size: 15000000, time: 1.651000

size: 25000000, time: 2.889000

size: 35000000, time: 3.932000

size: 45000000, time: 5.133000

size: 55000000, time: 6.804000

size: 65000000, time: 7.685000

size: 75000000, time: 8.767000

size: 85000000, time: 9.611000

size: 95000000, time: 11.469000

....................................................

## Измерение времени для пирамидальной сортировки HeapSort

....................................................

LOG FOR 1 THREAD

size: 100, time: 0.001000

size: 500, time: 0.001000

size: 1000, time: 0.003000

size: 2000, time: 0.007000

size: 5000, time: 0.044000

size: 10000, time: 0.196000

size: 50000, time: 6.439000

size: 75000, time: 14.843000

size: 100000, time: 27.768000

....................................................

LOG FOR 2 THREADS USING FOR

size: 100, time: 0.031000

size: 500, time: 0.086000

size: 1000, time: 0.148000

size: 2000, time: 0.283000

size: 5000, time: 0.701000

size: 10000, time: 1.482000

size: 50000, time: 10.976000

size: 75000, time: 20.106000

size: 100000, time: 33.852000

....................................................

LOG FOR 4 THREADS USING FOR

size: 100, time: 0.03000

size: 500, time: 0.081000

size: 1000, time: 0.147000

size: 2000, time: 0.293000

size: 5000, time: 0.704000

size: 10000, time: 1.427000

size: 50000, time: 9.036000

size: 75000, time: 15.695000

size: 100000, time: 23.624000

....................................................

### Анализ работы:

Алгоритм **сортировки слиянием** имеет сложность O(n log n). Он достаточно хорошо разделяется, поэтому его можно эффективно распараллелить с уменьшением времени обработки примерно обратно пропорциональному количеству процессов (при достаточно больших объемах массива).

В моем случае при размере массива >50млн удалось добиться ускорения в 2 и почти в 3 раза при использовании двух и четырех процессоров соответственно.

Алгоритм **пирамидальной сортировки** имеет примерную сложность **(n/2)·log2n. В отличие от первого алгоритма, данный способ сортировки практически невозможно (или не имеет смысла) распараллеливать, потому что каждая последующая итерация данной сортировки требует знания о результате из предыдущей итерации.**

**В моем случае алгоритм, распараллеленный на 4 процесса, начал увеличивать относительную скорость и обогнал сортировку 1 процессом примерно на 100тыс элементов массива. Я думаю, что при использовании массива, на порядки превышающего тренировочный, можно добиться увеличения скорости.**

**Если сравнивать алгоритмы между собой, то сортировка слиянием, безусловно, выигрывает у пирамидальной по скорости обработки, однако нуждается в дополнительной памяти для хранения частей массива.**

# **Вывод**

В ходе выполнения данной лабораторной работы я познакомилась с библиотекой для распараллеливания программ OpenMP, потренировалась сама распараллеливать алгоритмы, а так же повторила алгоритмы сортировки.