

Лабораторная работа №5 (ПОД №4)

Сортировка чисел на GPU. Свертка, сканирование, гистограмма.

Цель работы. Ознакомление с фундаментальными алгоритмами GPU: свертка (reduce), сканирование (blelloch scan) и гистограмма (histogram). Реализация одной из сортировок на CUDA. Использование *разделяемой* и других видов памяти. Исследование производительности программы с помощью утилиты nvprof (*обязательно отразить в отчете*).

Все входные-выходные данные являются бинарными и считываются из **stdin** и выводятся в **stdout**.

Входные данные. В первых четырех байтах записывается целое число n -- длина массива чисел, далее следуют n чисел типа заданного варианта.

Выходные данные. В бинарном виде записывают n отсортированных по возрастанию чисел.

Пример входных-выходных данных. Десять чисел типа `int`, от 0 до 9.

Входной файл (`stdin`), hex:

```
0A000000 00000000 09000000 08000000 07000000 06000000 05000000
04000000 03000000 02000000 01000000
```

Выходной файл (`stdout`), hex:

```
00000000 01000000 02000000 03000000 04000000 05000000 06000000
07000000 08000000 09000000
```

Вариант 1. Битоническая сортировка.

Требуется реализовать битоническую сортировку для чисел типа `int`.

Должна быть реализована адаптивная операция битонического слияния. Если данные помещаются в разделяемую память, то взаимодействие идет через неё, если нет, то через глобальную память (т.е. необходимо реализовать несколько вариантов ядра).

Ограничения: $n \leq 256 * 10^6$

Вариант 2. Сортировка подсчетом. Диапазон от 0 до $2^{24} - 1$.

Требуется реализовать сортировку подсчетом для чисел типа `int`.

Должны быть реализованы:

- Алгоритм гистограммы, с использованием атомарных операций.
- Алгоритм сканирования для любого размера, с рекурсией и бесконфликтным использованием разделяемой памяти.

Ограничения: $n \leq 135 * 10^6$

Вариант 3. Сортировка подсчетом. Диапазон от 0 до 255.

Требуется реализовать сортировку подсчетом для чисел типа uchar.

Должны быть реализованы:

- Алгоритм гистограммы, с использованием атомарных операций и разделяемой памяти.
- Алгоритм сканирования, с бесконфликтным использованием разделяемой памяти.

Ограничения: $n \leq 537 * 10^6$

Пример:

| Входной файл (stdin), hex | Выходной файл (stdout), hex |
|--|-------------------------------|
| 0A 00 00 00 01 02 03 01 02 03 01 02 03 04 | 01 01 01 02 02 02 03 03 03 04 |

Вариант 4. Сортировка чет-нечет.

Требуется реализовать блочную сортировку чет-нечет для чисел типа int.

Должны быть реализованы:

- Алгоритм чет-нечет сортировки для предварительной сортировки блоков.
- Алгоритм битонического слияния, с использованием разделяемой памяти.

Ограничения: $n \leq 16 * 10^6$

Вариант 5. Сортировка чет-нечет с предварительной битонической сортировкой.

Требуется реализовать блочную сортировку чет-нечет для чисел типа int.

Должны быть реализованы:

- Алгоритм битонической сортировки для предварительной сортировки блоков.
- Алгоритм битонического слияния, с использованием разделяемой памяти.

Ограничения: $n \leq 16 * 10^6$

Вариант 6. Карманная сортировка с битонической сортировкой в каждом кармане.

Требуется реализовать карманную сортировку для чисел типа float.

Должны быть реализованы:

- Алгоритм гистограммы, с использованием атомарных операций.
- Алгоритм свертки для любого размера, с использованием разделяемой памяти.
- Алгоритм сканирования для любого размера, с использованием разделяемой памяти. (Можно воспользоваться библиотекой Thrust)

- Алгоритм битонической сортировки для карманов.

Ограничения: $n \leq 100 * 10^6$

Пример:

| Входной файл (stdin), hex | Выходной файл (stdout), hex |
|---|--|
| 0A000000 00000000 00001041 00000041 0000E040 0000C040 0000A040 00008040 00004040 00000040 0000803F | 00000000 0000803F 00000040 00004040 00008040 0000A040 0000C040 0000E040 00000041 00001041 |

Комментарий: 0.00 9.00 8.00 7.00 6.00 5.00 4.00 3.00 2.00 1.00

Вариант 7. Карманная сортировка с чет-нечет сортировкой в каждом кармане.

Требуется реализовать карманную сортировку для чисел типа float.

Должны быть реализованы:

- Алгоритм гистограммы, с использованием атомарных операций.
- Алгоритм свертки для любого размера, с использованием разделяемой памяти. (Можно воспользоваться библиотекой Thrust)
- Алгоритм сканирования для любого размера, с использованием разделяемой памяти.
- Алгоритм чет-нечет сортировки для карманов.

Ограничения: $n \leq 100 * 10^6$

Пример:

| Входной файл (stdin), hex | Выходной файл (stdout), hex |
|---|--|
| 0A000000 00000000 00001041 00000041 0000E040 0000C040 0000A040 00008040 00004040 00000040 0000803F | 00000000 0000803F 00000040 00004040 00008040 0000A040 0000C040 0000E040 00000041 00001041 |

Комментарий: 0.00 9.00 8.00 7.00 6.00 5.00 4.00 3.00 2.00 1.00

Вариант 8. Поразрядная сортировка.

Требуется реализовать поразрядную сортировку для чисел типа uint.

Должны быть реализованы:

- Алгоритм сортировки через префиксные суммы для одного битового разряда.
- Алгоритм сканирования для любого размера, с рекурсией и бесконфликтным использованием разделяемой памяти.

Ограничения: $n \leq 128 * 10^6$

Вариант на “два”. Сортировка подсчетом.

Вариант №2, с использованием алгоритма сканирования из библиотеки Thrust.