Московский Государственный Университет

имени М.В. Ломоносова

Курс: «Параллельное программирование и суперкомпьютеры»

Расписание сети сортировки

**Выполнил: Сактаганов Нуржан**

**Дата подачи: 23.10.2017**

**Москва 2017**

## Постановка задачи

Разработать последовательную программу вычисления расписания сети сортировки, числа использованных компараторов и числа тактов, необходимых для её срабатывания привыполнении на **n** процессорах. Число тактов сортировки при параллельной обработке не должно превышать числа тактов, затрачиваемых четно-нечетной сортировкой Бетчера.

Параметр командной строки запуска: n. n > 0 – количество элементов в упорядочиваемом массиве, элементы которого расположены на строках с номерами [0…n-1].

Формат команды запуска: **./program n**

Требуется:

1. вывести в файл стандартного вывода расписание и его характеристики в представленном далее формате;
2. обеспечить возможность вычисления сети сортировки для числа элементов 1<=n<=10000;
3. предусмотреть полную проверку правильности сети сортировки для значений числа сортируемых элементов 1<=n<=24;
4. формат файла результата:

n 0 0

cu0 cd0

cu1 cd1

…

cun\_comp-1 cdn\_comp-1

n\_comp

n\_tact

Здесь:

n 0 0 – число сортируемых элементов, ноль, ноль.

cui cdi – номера строк, соединяемых i-м компаратором сравнения перестановки.

n\_comp – число компараторов

n\_tact– число тактов сети сортировки

## Метод решения

Метод построения основан на рекурсивном алгоритме. В алгоритме предполагается, что n — степень двойки. Расписания сортировки для массива из n элементов состоит из расписания сортировки отдельных половинок массива, и затем четно-нечетного объединения этих половинок.

Алгоритм построения легко расширяется для произвольного n > 0. Для этого находится наименьшее число k >= n, которое является степенью двойки. Строится расписание, и из полученных компараторов вида (a, b) отбрасываются все такие, для которых b >= n. Корректность такого отбрасывания можно обосновать тем, что можно предположить, что на всех позициях i, где n <= i < k, находятся заведомо большие элементы (например, MAX\_INT) и обмен никогда не будет происходить.

В программе реализован алгоритм, строящий расписание для произвольного n > 0.

## Метод проверки

Проверка каждого расписания для 1 <= n <= 24 производится принципом нулей и единиц. Т.е. берутся все возможные комбинации нулей и единиц заданной длины n – а их всего два в степени n. Над каждой такой комбинацией производятся операции сравнения и обмена согласно проверяемому расписанию. После выполнения расписания над текущей комбинацией, проверяется, что она отсортирована. Расписание считается корректной, если все комбинации нулей и единиц заданной длины в итоге окажутся отсортированы в результате выполнения расписания.

Эту проверку можно запустить передав третий необязательный параметр. Например:

**program 0 k**

Здесь k указывает до какой длины (включительно) проверяется алгоритм построения расписания.