Задача A2 отчёт

Ссылка на публичный репозиторий:

https://github.com/samstarkov/set3-algo/

set_a2i.cpp – код, который засылался в codeforces, для проверки работоспособности алгоритма

ID посылки codeforces: 349192793

Реализация на полуинтервалах [l; r).

Использовался параметр переключения на INSERTION SORT равный 15 (как по условию в codeforces).

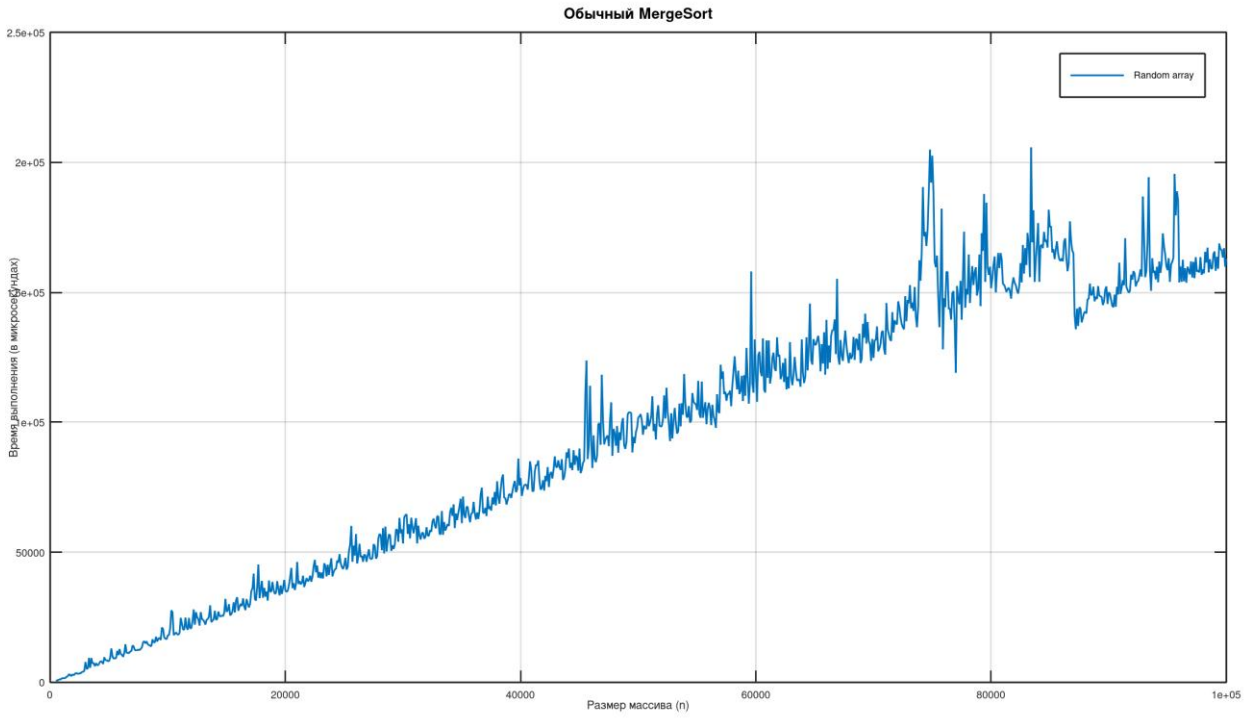
set_a2_addition.cpp; SortFuncs.h; ArrayGenerator.h; SortTester.h; - файлы содержащие код для получения данных для дальнейшего анализа (составления графиков).

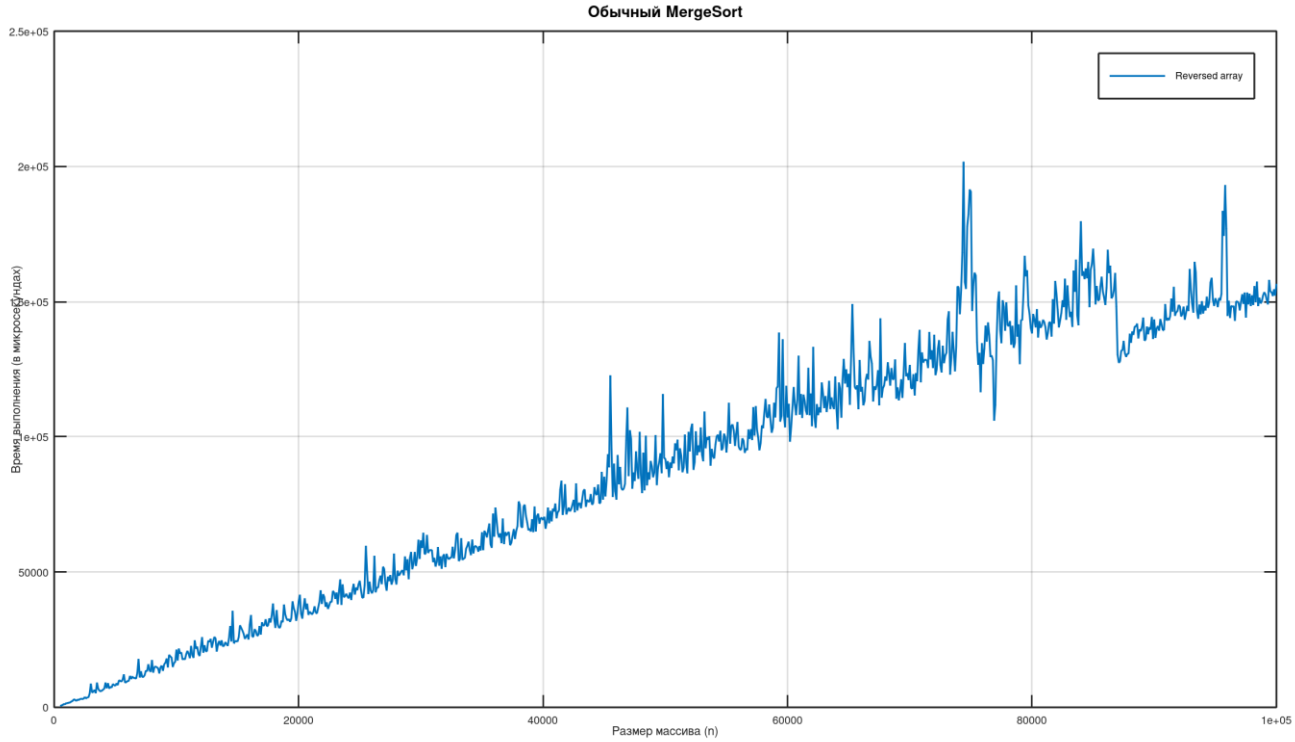
MergeSortRandom.txt; MergeSortReversed.txt; MergeSortAlmostSorted.txt; - файлы с результатами обычного MergeSort для массивов различных видов. Каждый файл содержит два столбца, первый из которых – значения n, второй – результат замера времени (в макросекундах).

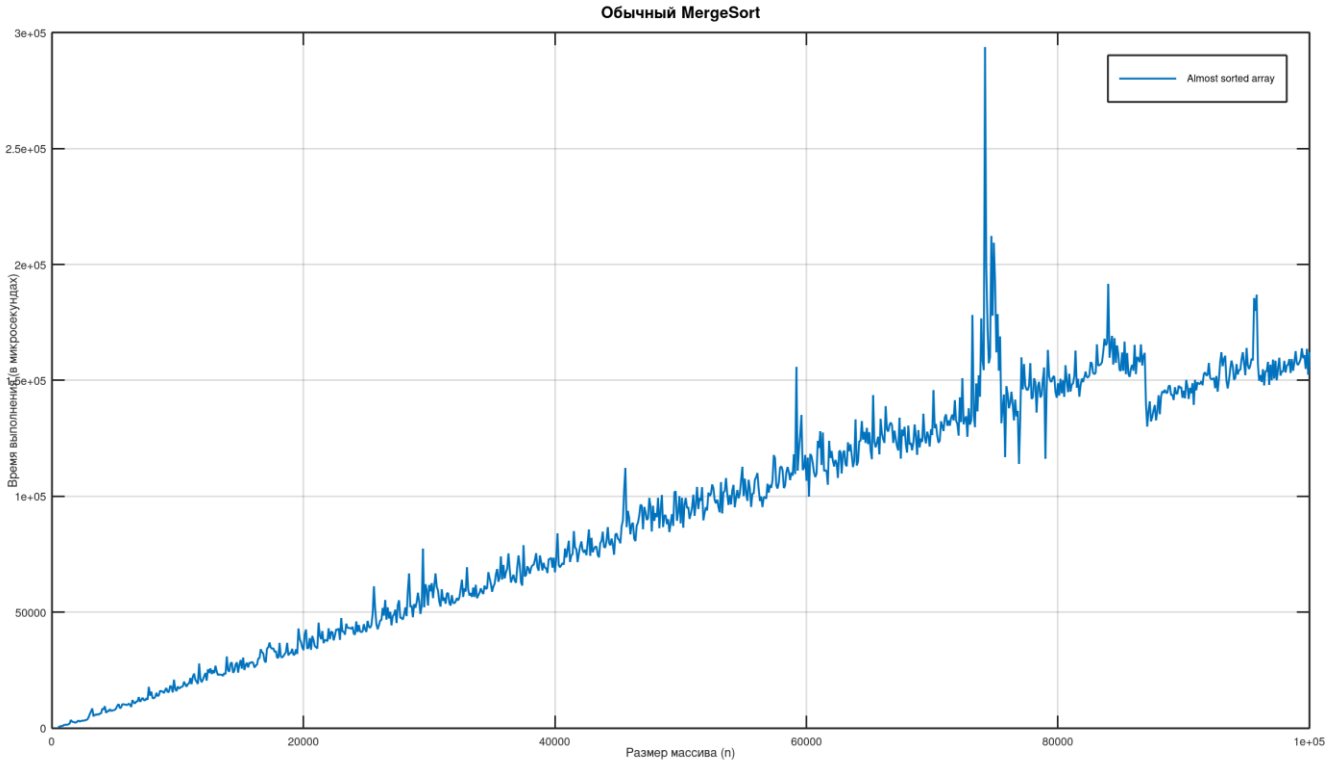
HybridMergeSortRandom.txt; HybridSortReversed.txt; HybridSortAlmostSorted.txt; - файлы с результатами HybridMergeSort для массивов различных видов. Каждый файл содержит шесть столбцов, первый из которых – значения n, а остальные 5 – результаты замера времени для различных *threshold* (5, 10, 20, 30 и 50 соответственно).

Далее, в Octave (MatLab) написан код, для представления графиков, необходимых по условию задачи (файл **set3_a2_plots.m**):

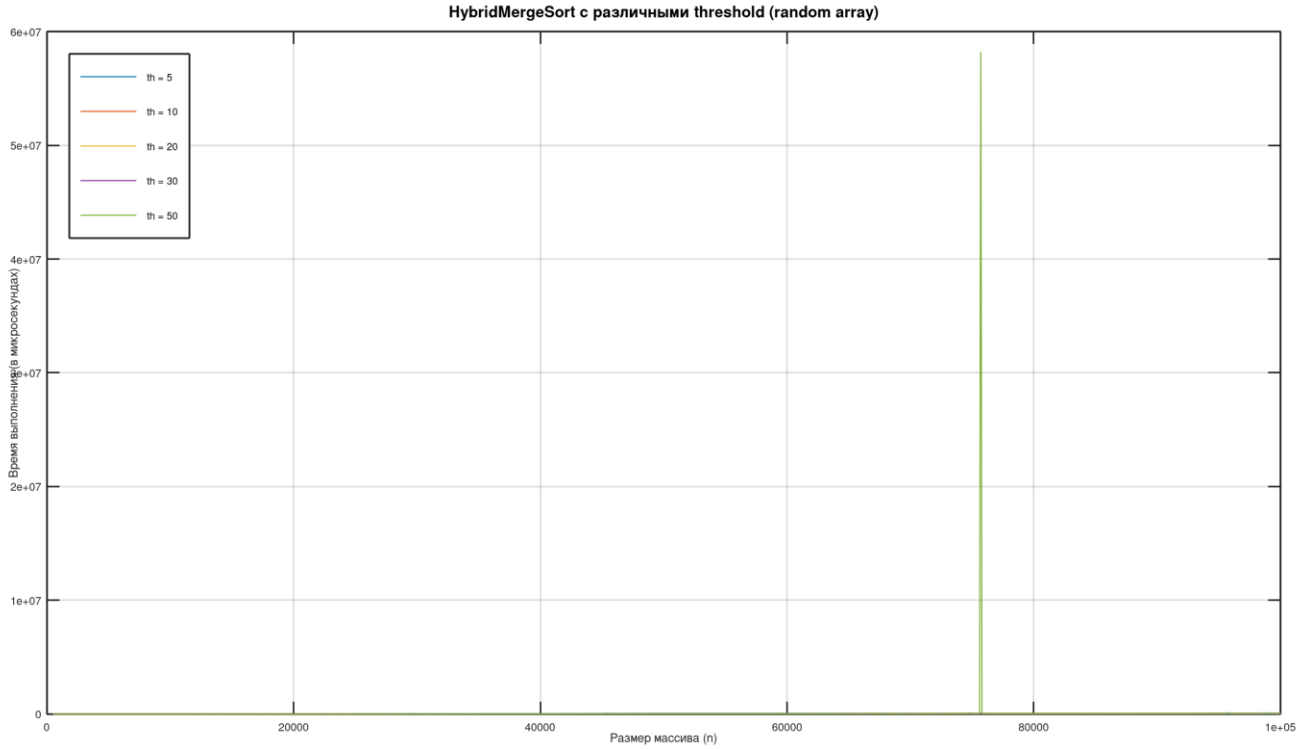
Работа обычного MergeSort:



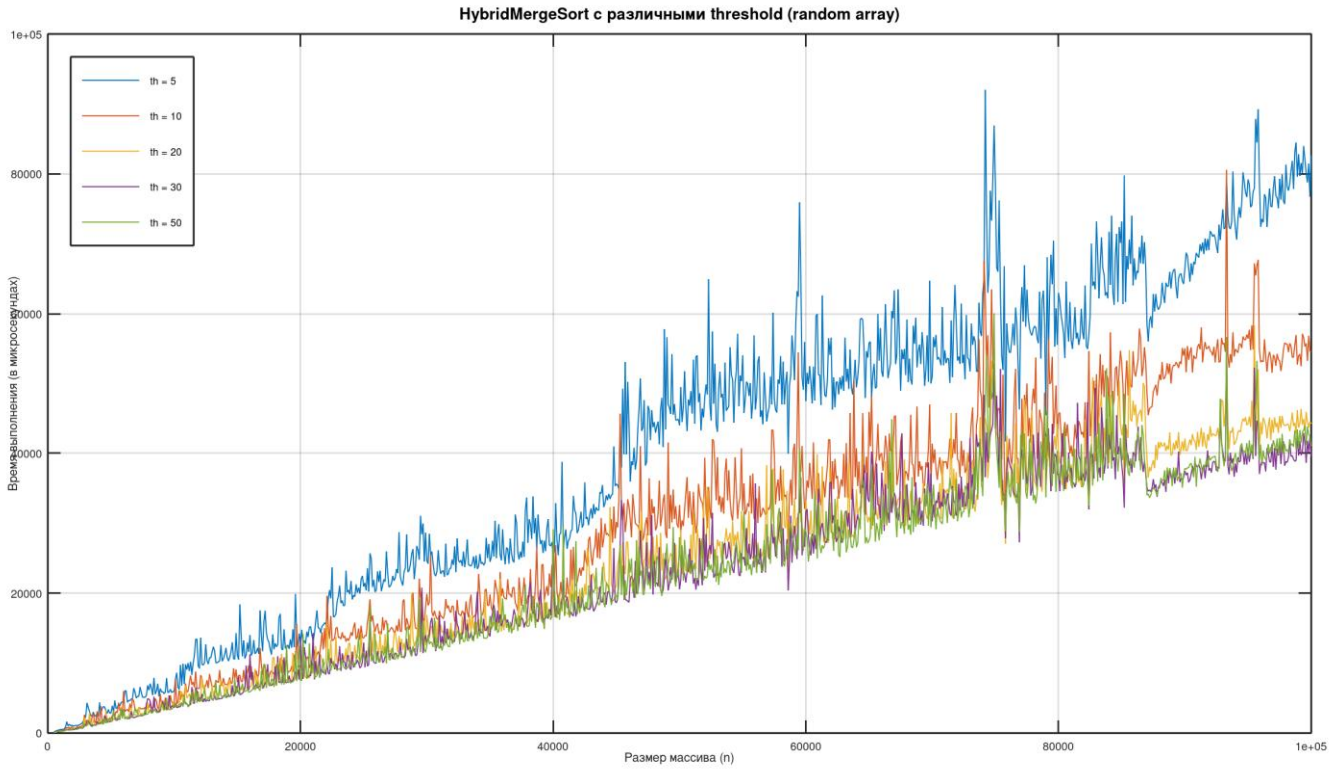

Обычный MergeSort
Reversed array
Время выполнения (в микросекундах)
Размер массива (n)
2.5e+05
2e+05
1e+05
50000
0
0
20000
40000
60000
80000
1e+05

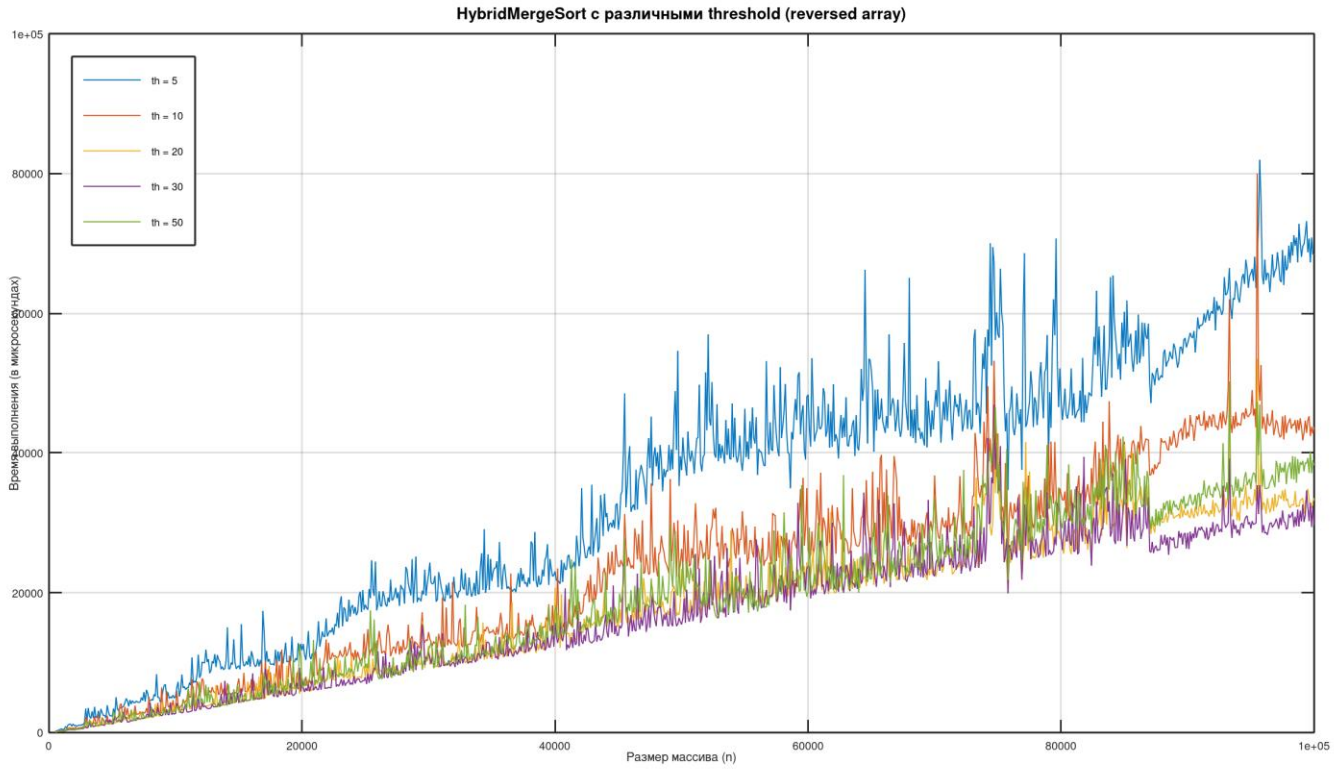

Обычный MergeSort
Almost sorted array
Время выполнения (в микросекундах)
Размер массива (n)
3e+05
2.5e+05
2e+05
1e+05
50000
0
0
20000
40000
60000
80000
1e+05

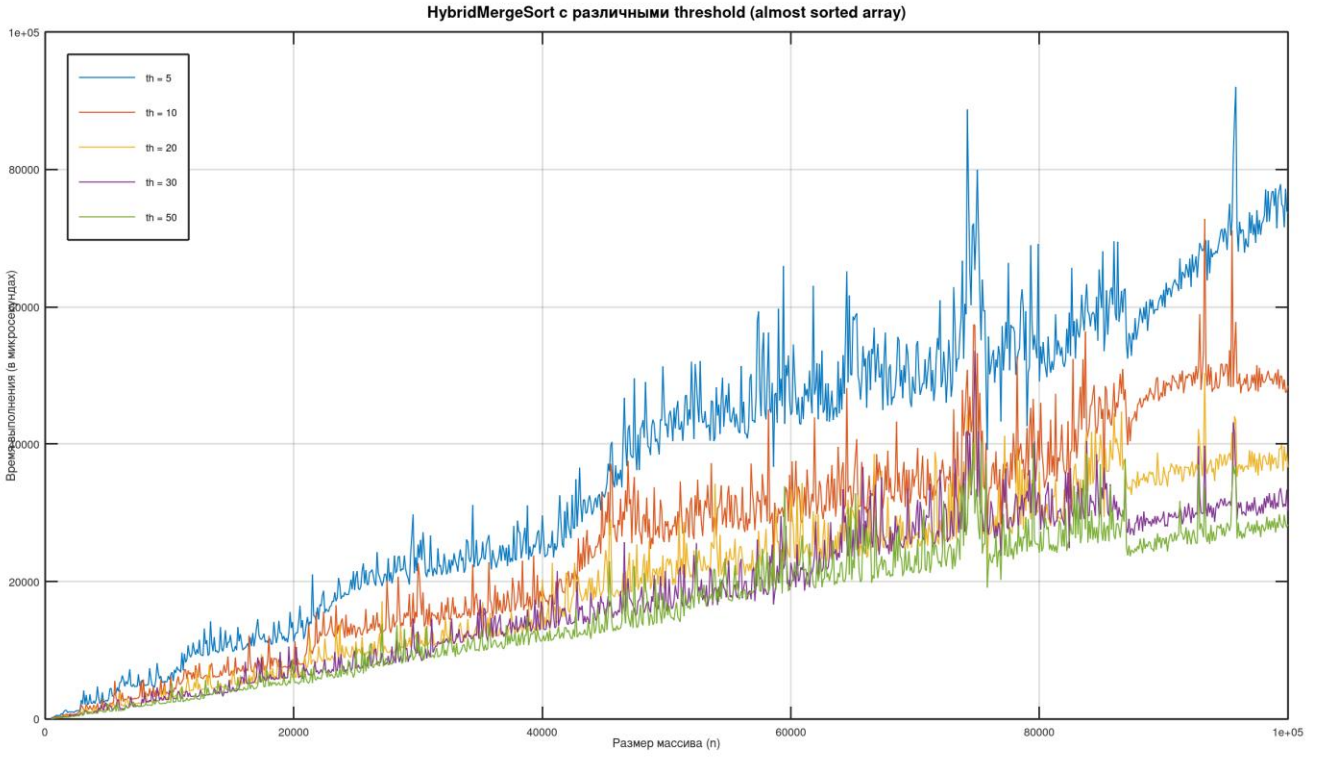
Работа HybridMergeSort:



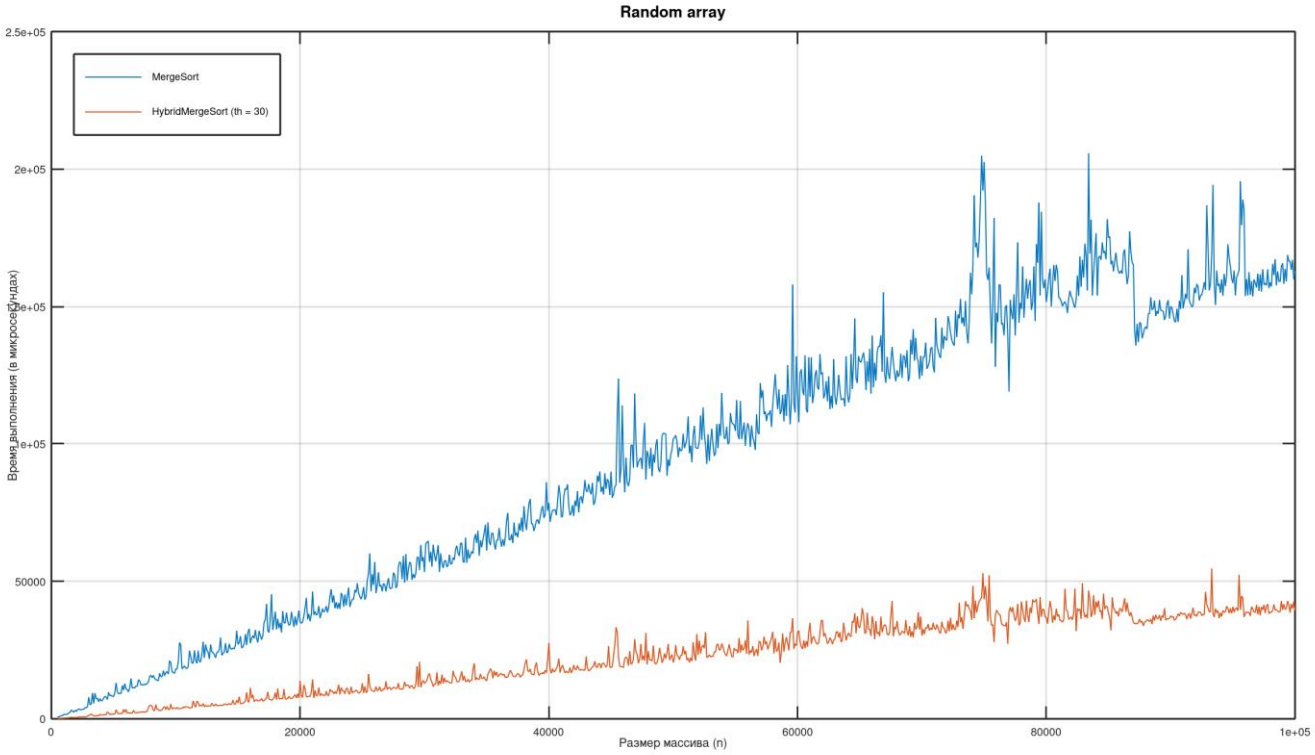
Чем вызван такой скачок, я предположить не могу, поэтому я не учитывал значения для n = 75700, чтобы хотя бы был виден график (приложен следующим), а так для threshold = 50 было 5.82012e+07 микросекунд:

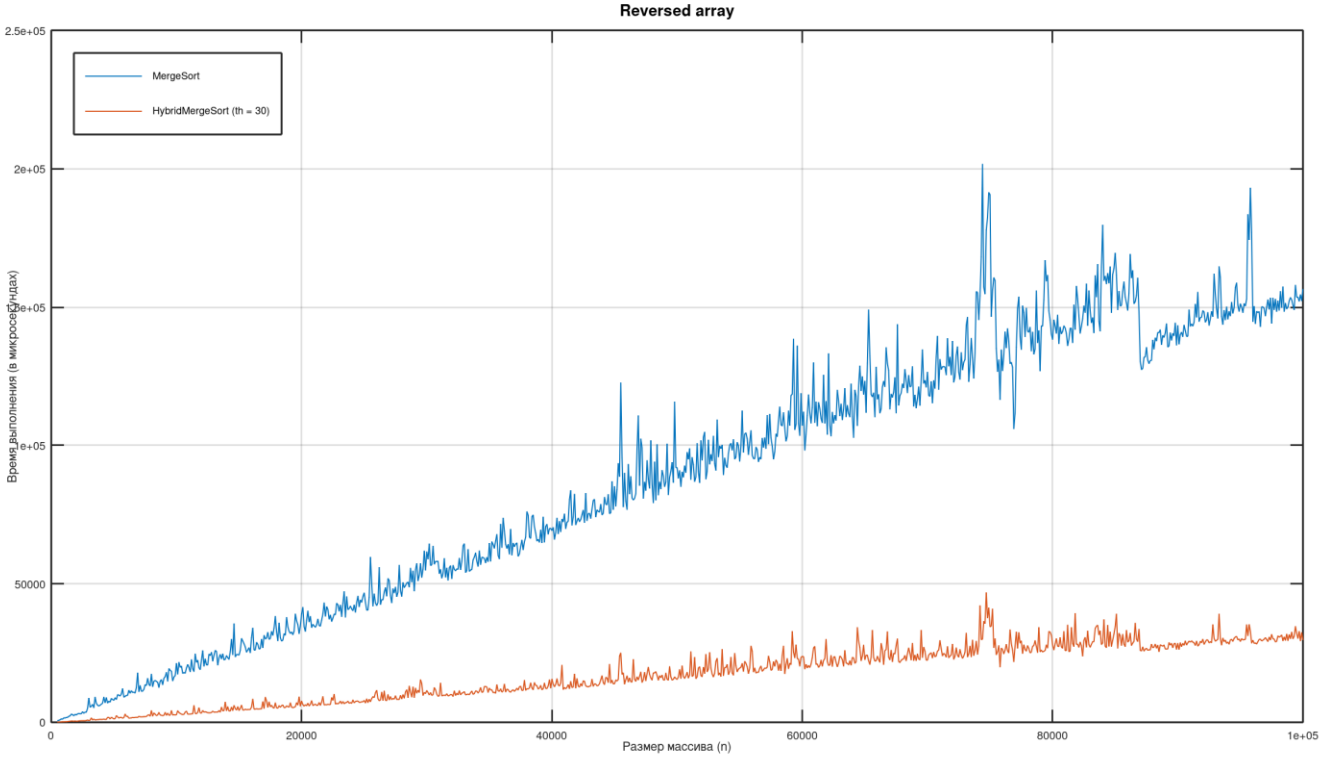


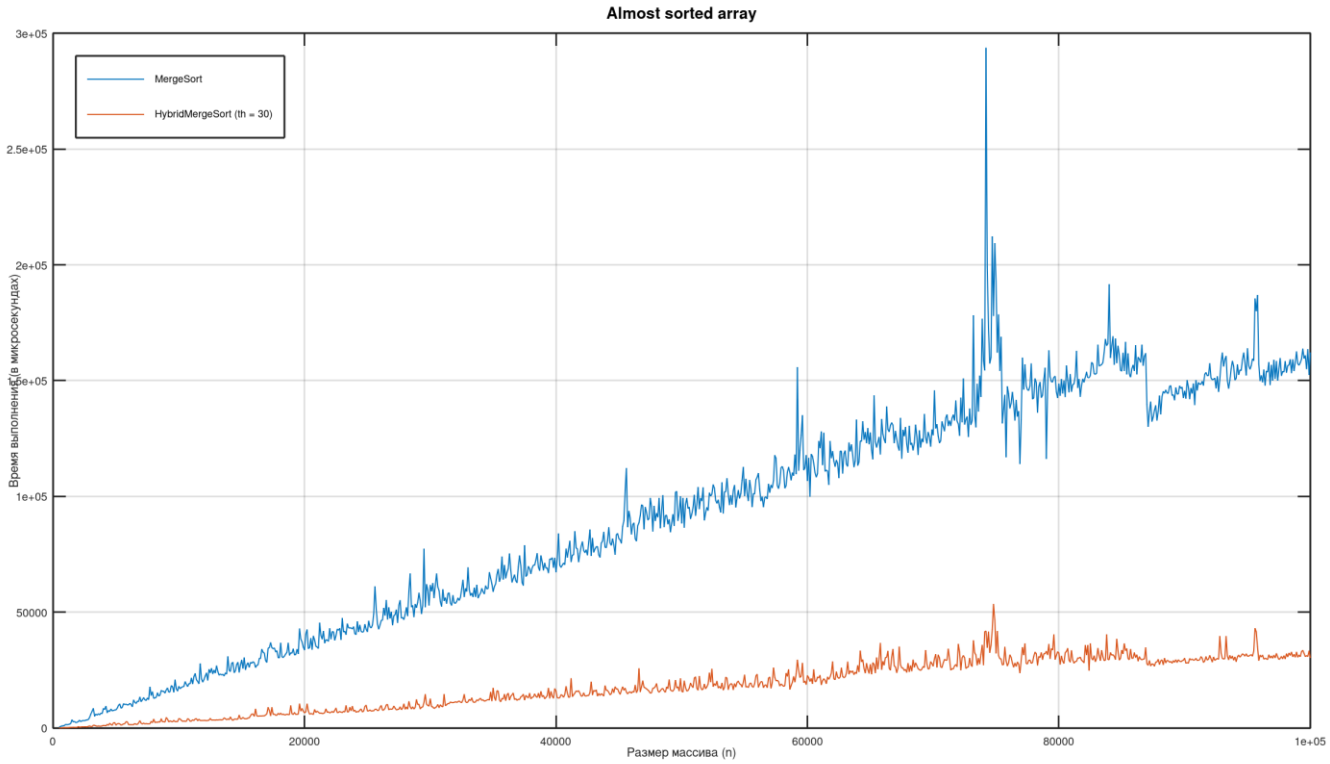

HybridMergeSort с различными threshold (reversed array)
th = 5
th = 10
th = 20
th = 30
th = 50
Время выполнения (в микросекундах)
Размер массива (n)
0
20000
40000
60000
80000
1e+05


HybridMergeSort с различными threshold (almost sorted array)
th = 5
th = 10
th = 20
th = 30
th = 50
Время выполнения (в микросекундах)
Размер массива (n)
0
20000
40000
60000
80000
1e+05

Сравнение эффективности обычного MergeSort и HybridMergeSort:







Из-за более эффективной работы алгоритма InsertionSort для небольших массивов получаем более быструю работу алгоритма HybridMergeSort благодаря переходу на InsertionSort при получении на очередном этапе рекурсии небольшого массива (размера <= threshold). От самого параметра threshold также зависит время работы алгоритма (в моём случае наиболее эффективно алгоритм работает при threshold = 30 для случайного и обратно отсортированного массивов, а для почти отсортированных эффективнее при threshold = 50).

При этом время работы также зависит от вида самого массива. Например, исходя из полученных графиков видно, что reversed и almost sorted массивы сортируются обычно быстрее, чем random массивы.

Исходя из полученных результатов, можно сделать вывод, что HybridMergeSort действительно работает эффективнее, чем MergeSort, особенно при правильно подобранном threshold.