PS 703106
Exercise 8

Group 4:
• Jonas Boutelhik
• Michael Thöni
• Thomas Urban

Hardware:
1. GeForce GTX 860M
 1.1 Hardware version: OpenCL 1.2 CUDA
 1.2 Software version: 418.39
 1.3 OpenCL C version: OpenCL C 1.2
 1.4 Parallel compute units: 5

Assignment - Count Sort:
Implement list_sort of the first assignment in OpenCL using the prefix sum
implementation of the second assignment.
o Implement the histogram step (first step of count sort) using OpenCL
  parallelisation strategy:
   n batches for n workgroups calculating n histogramms (< 128 histograms no doubt)
   and have then a reduction step on these -> 1 step at most all in all probably same or greater
   number of global memory access so not worth the logic overhead
o Implement the element copying (third step of count sort) using OpenCL
  128 worker running the loop (from 1 to n) and writing data in resulting array.
  The use of local memory is possible: Having a local_memsize sized batch (N) of start data
  and then load them into memory (in parallell by the 128 (or N superfluous) workers) and
  loop through the batches of start overhead in opencl host logic.
o Write a test program countsort_bench.c, which takes the same parameters as
the program of the first assignment. However, instead of printing the names, it
should just measure and print the time required for sorting.

The whole OCL program runs with 5 kernel runs:
  1) histogram step
  2, 3 and 4) are for the prefix sum calculation according to prefixglobal from last proseminar
  5) copying step

To compare the Count Sort implementations of SEQ and OCL the program was executed with
element numbers ranging from 500-20.000. The step width was 500 and the seed 42. As seen in
Figure 1 our OCL implementation is about 20 times slower than the SEQ version at a N of 20.000.
Due to lack of time, we were unable to find the reason for the poor performance of to code and to
optimize it.

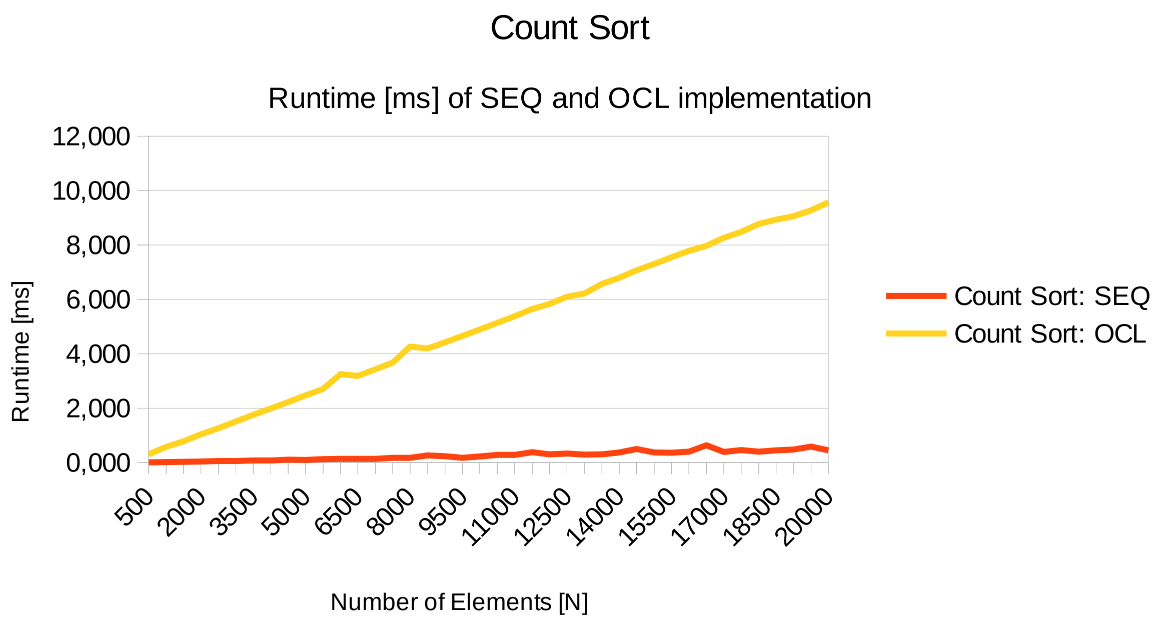

Fig. 1: Runtime performance of comparing the Count Sort for SEQ and OCL implementation