Radix Sort GPU Lab project WS2021/2022

Lab Course High Performance Programming with Graphic Cards (**Gasim Mammadov**)

By Group4

Tobias Brodmann, Wenxin Wang, David Bauer

Radix sort is a non-comparative sorting algorithm. It avoids comparison by creating and distributing elements into buckets according to their radix. For elements with more than one significant digit, this bucketing process is repeated for each digit, while preserving the ordering of the prior step, until all digits have been considered. For this reason, radix sort. (Wikipedia)

It is a O(n) time complexity sorting algorithm which is exceptionally fast. Since its non-comparative nature we can only sort specific things like integers or strings.

The nature of radix sort is not a parallel one – but there exist several ways to achieve some parallel execution.

We chose a nice way that is also in use with the Open MPI library and described here: https://himnickson.medium.com/parallel-radix-sort-algorithm-using-message-passing-interface-mpi-31b9e4677fbd

Our parallel Radix Sort works by first sorting the data on each GPU core according to the digit value for each key digit. (using local sort)

That is done by all the cores on a subset of the data at the same time.

Then, determine which data of which sorted subset must be send where to the local data of other cores in order to have the distributed list sorted globally and by digit.

Algorithm flow:

Input: rank (rank of the processor), L (portion of the distributed data held by this processor)

Output: the distributed data sorted across the processors with the same amount of data for each processor

1. For each keys digit i where i varies from the least significant digit to the most significant digit:

2. Use **Counting So**rt to sort L according to the i th keys digit

3. Share information with other processors to figure out which local data to send where and what to receive from which processors in order to have the distributed array sorted across processors according to the i th keys digit

4. Proceed to these exchanges of data between processors

Here this is done to some example data:

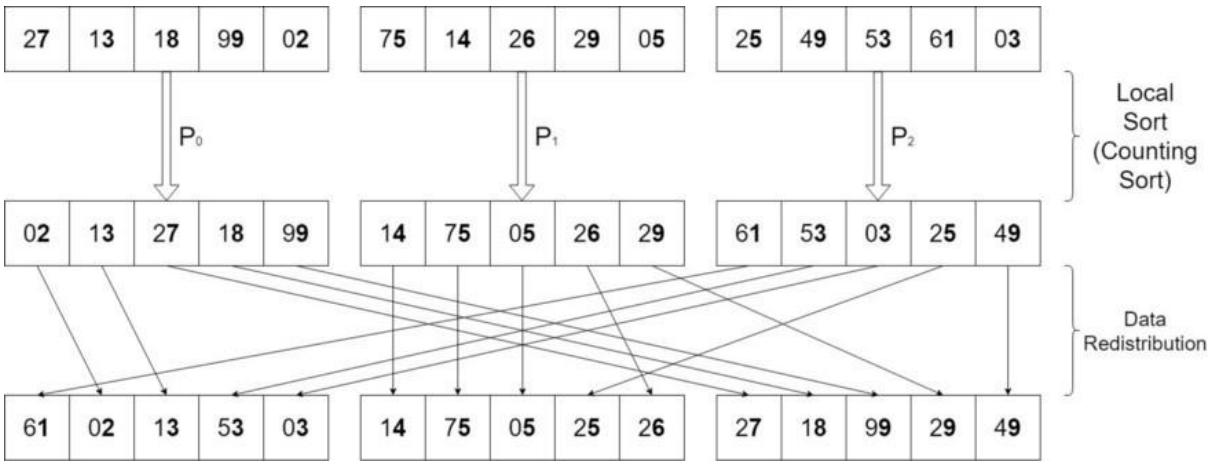


Image by Nickson Joram: LSD based Parallel Radix Sort

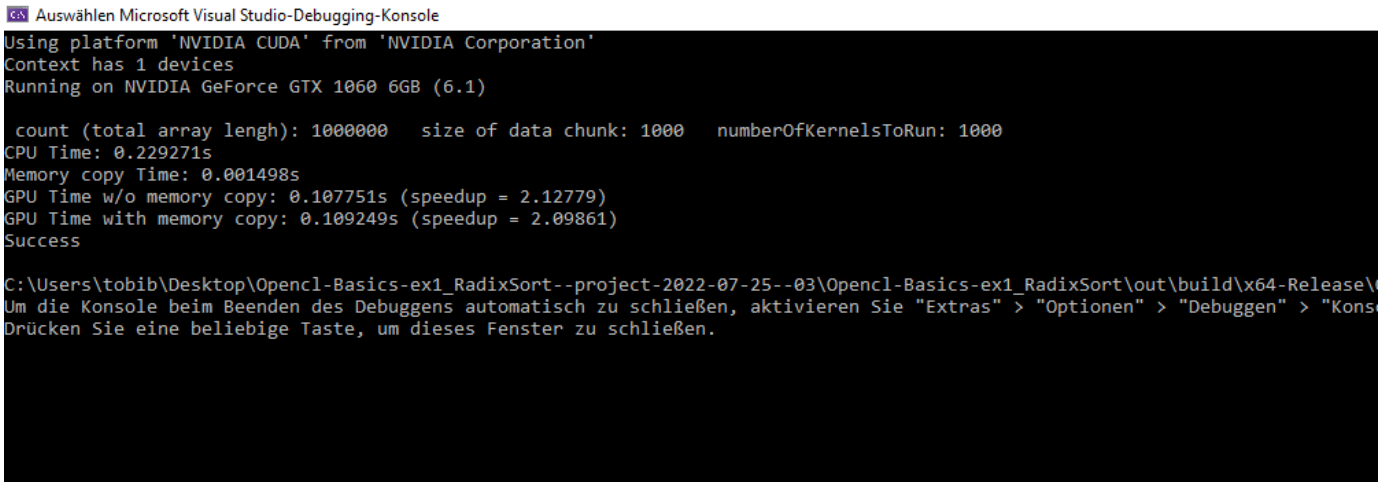
The counting sort of the subsets of the data happens in parallel on GPU cores.

The data redistribution we chose to do on the host.
Our implementation may be better understood with reading
https://stackoverflow.com/questions/26206544/parallel-radix-sort-how-would-this-implementation-actually-work-are-there-some/26229897#26229897

(creating of histogram, offsets, prefix sum)

We repeat the process for each digit and we end up very fast with the sorted the input data.

In our implementation we reached a quite significant speedup compared to running radix sort (non parallel on the CPU).



(speed up on a 1060 TI in release on an Amd Ryzen 5 1600)

This is one of the highest speedups we had. On Average the speed up on this set up was between 1.5 and 1.9.

All in all it was a fun project and we now have a way better understanding of this way of doing radix sort in parallel and working with GPUs.

References/Sources:

https://himnickson.medium.com/parallel-radix-sort-algorithm-using-message-passing-interface-mpi-31b9e4677fbd

https://en.wikipedia.org/wiki/Radix_sort