Prof. G. Zachmann
D. Mohr

University of Bremen
School of Computer Science
CGVR Group
April 26, 2013

Summer Semester 2013

Assignment on Massively Parallel Algorithms - Sheet 2

Due Date 08. 05. 2013

Exercise 1 (Reverse Array (single block), *2 Punkte*)

Starting from the `reverse_array_single` template, and given an input array $\{a_0, a_1, \ldots, a_{n-1}\}$ in pointer `d_a`, store the reversed array $\{a_{n-1}, a_{n-2}, \ldots, a_0\}$ in pointer `d_b`. Launch only one thread block, to reverse an array of size `N = numThreads = 256` elements.

All you have to do is implement the body of the kernel `reverseArrayBlock()`. Each thread moves a single element to reversed position:

a) Read input from array `d_a`
b) Store output in reversed location in array `d_b`

Exercise 2 (Reverse Array (multiblock), *2 Punkte*)

Starting from the `reverse_array_multi` template, and given an input array $\{a_0,\ a_1,\ ...,\ a_{n-1}\}$ in array `d_a`, store the reversed array $\{a_{n-1},\ a_{n-2},\ ...,\ a_0\}$ in array `d_b`. Launch multiple 256-thread blocks; to reverse an array of size N, you need N/256 blocks.

a) Compute the number of blocks to launch
b) Implement the kernel reverseArrayBlock()

Note that now you must compute both the reversed location within the block, as well as the reversed offset to the start of the block.

Exercise 3 (Fractals, *6 Punkte*)

Framework `fractal_zoomer` provides a reference implementation of a Mandelbrot generator.

a) Convert the reference implementation to a massively parallel GPU kernel and compare the results to the reference implementation. (You can (re-)use the source code from the lecture webpage, if you want). You only have to write the kernel function `fractal_gpu`.
   Compute the root mean square (RMS) error

$$E_{RMS} = \sqrt{\frac{1}{w \cdot h} \sum_{\mathbf{x}=\mathbf{0}}^{(w,h)} \left(I_{CPU}(\mathbf{x}) - I_{GPU}(\mathbf{x})\right)^2} \qquad (1)$$

   between the CPU and GPU implementation. What does the result mean?
   Hint: You can switch between the CPU and GPU version by pressing the space key.

b) What happens when you zoom in very deeply? Examine a zoom in on the main antenna (Fig.1 arrow 1) that sticks out to the left, and another one into he upper antlers (Fig.1 arrow 2) growing out from the top of the main cardioid. Can you explain the different effects when you approach the limits of floating point precision?

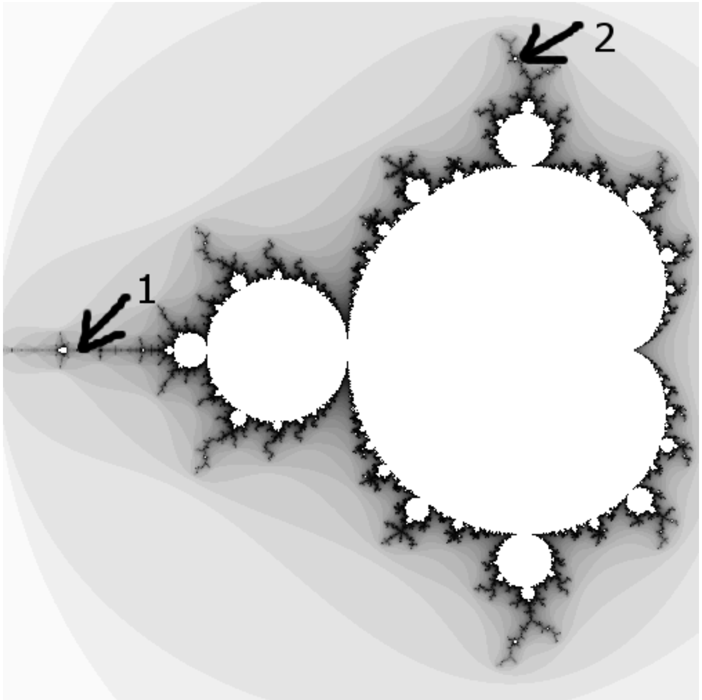

Figure 1: Mandelbrot Points of Interest

c) Vary the block size of the kernel call, e.g. horizontal or vertical stripes, rectangles and squares of different sizes (use powers of two for simplicity). How does that influence running times?