Lab B Part 2

Anton Bengtsson, Deepak Singh

May 2018

1 Getting Started

1.1 Process on s1 and s2

The process function returned the answer 73i32.

1.2 Testing on generated data

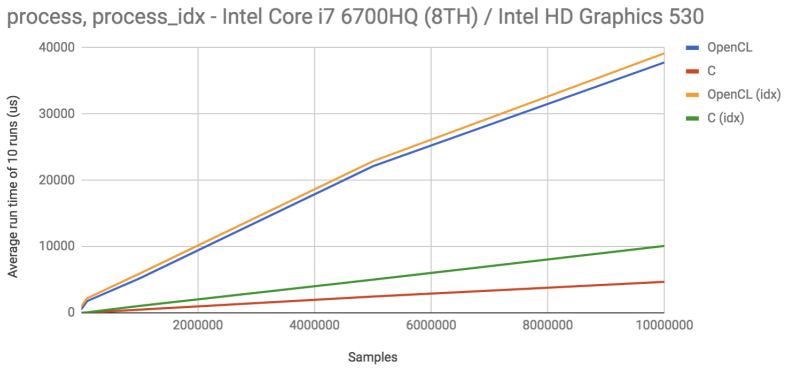

Figure 1: Performance results for process and process_idx

1.3 Neutral elements

Since the if statement goes to its default state for $f_2 = true$ and or only will be set to true if f_1 is set to true (f_2 is false as a result of the if statement), false has to be the neutral element since the or operator would not have any meaning otherwise.

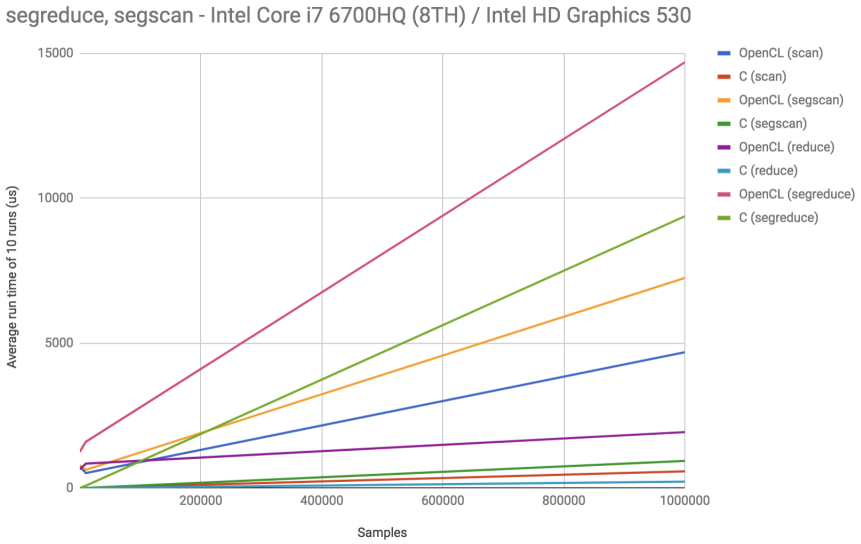

Figure 2: Performance comparison of segmented functions

1.4 Segmented functions

When bench-marking the segmented versions of the functions scan and reduce (figure 2) it becomes clear that not all methods of implementation will scale well when done in parallel. All benchmarks made on sequential code is faster since the overhead is so big on small, parallel work.

2 Monte Carlo

2.1 Computation Times

From benchmarks made of calculating integrals and pi with the Monte Carlo algorithm it is possible to see that the algorithm for pi scales better with sequential code and that the integral scales better in parallel. This can be seen in figure 3.

2.2 Sobol

The approximation converges faster with the sobol generated random numbers.

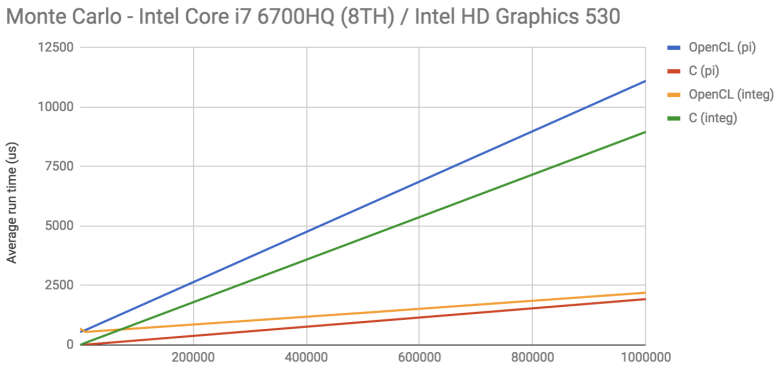

Figure 3: Performance results for monte carlo approximations of pi and integral

3 2D Ising simulation

From figure 4 it is very easy to see the power of parallel execution, as the performance gets exponentially better when the GPU can kick in to help.

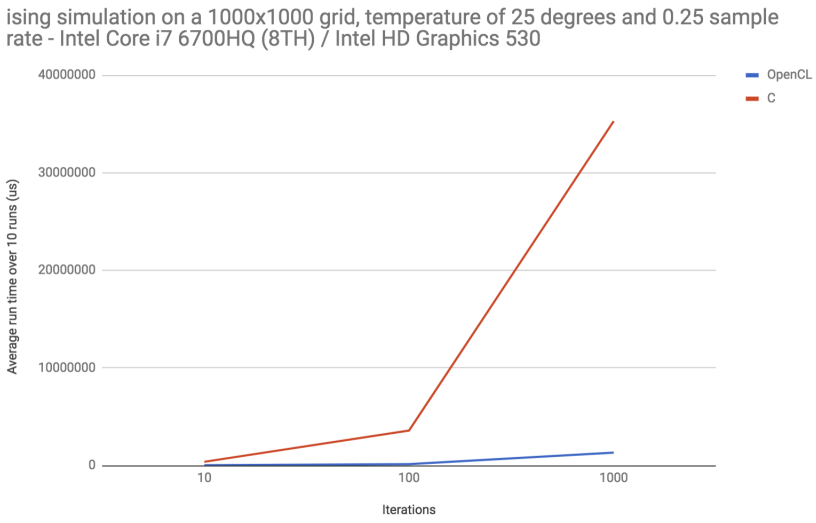

Figure 4: Performance results for the ising simulation