Università della Svizzera italiana

Facoltà di scienze informatiche

High Performance Computing **2017**

Student: Timon Willi

Solution for Assignment 4

Due date: 7 November 2017, 13:30

1. 1. Implementing linear algebra functions and the stencil operators *(40 Points)*

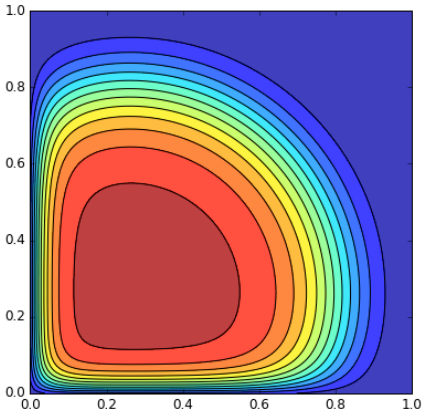

Figure 1. Result after Sequential Run with 128x128 100 0.01

2. Adding OpenMP to the mini-app *(60 Points)*

2.1. Did your changes make any performance improvement?

As we can see, the parallelization of the linear algebra script has lead to significant improvement. For the size 128x128, the time is only 55% of what it took for the serial version and for the 256x256 we halve the time it takes to run.

Table 1. Performance Comparison of LinAlg Parallelization, 10 Threads

	Serial/Serial	Parallel/Serial
128x128	1.32924	0.728149
256x256	9.03888	4.63804

2.2. Testing

The performance is decreasing constantly with increasing the number of threads. However, it is flattening towards 10 threads, suggesting that using more than 10 threads will not bring more improvement. Using more threads might even lead to more overhead and thus decreasing the performance. We can also see that the smaller the size, the less it scales. This has to do with the overhead coming with OpenMP. If the columns and rows are bigger, we can do more calculations relative to the amount of parallel regions that we needed to create. Thus it scales better. I have omitted 1024x1024 as it takes a lot of time to run.

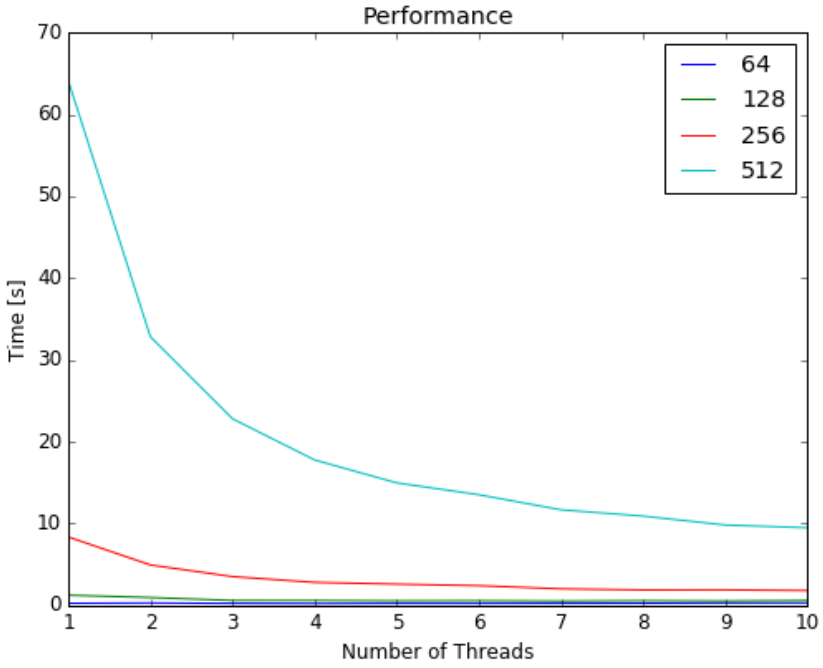

Figure 2. Time Comparison of different sizes with different Nr. of Threads, ran on node 27

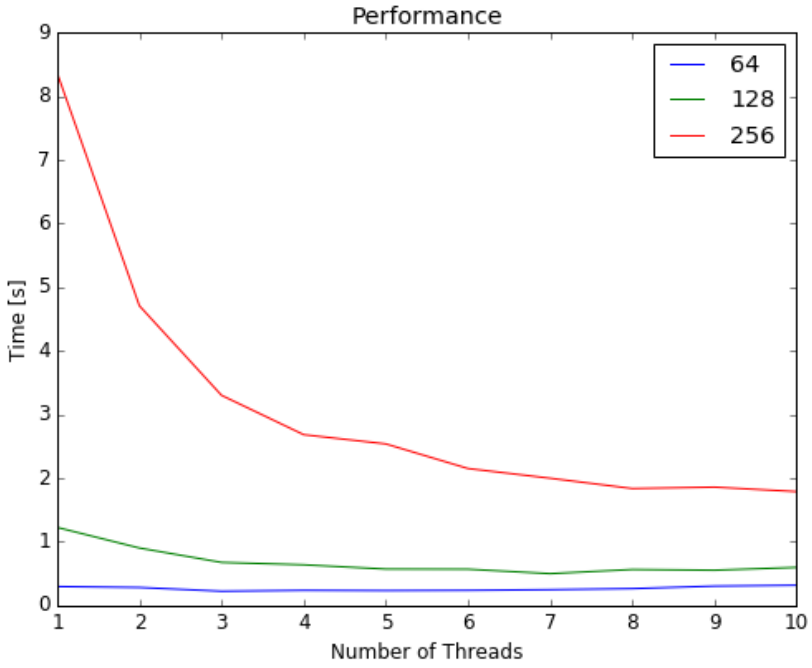

Figure 3. Time Comparison of different sizes with different Nr. of Threads, ran on node 27