ICS-E4020: Week 3 - Sorting/ImageSegmentation

Néstor Castillo García (472081)
nestor.castillogarcia@aalto.fi

May 4, 2015

1 Parallel Merge Sort Algorithm

1.1 Description

In this task a parallel version of merge sort was implemented. In the first step the data is divided into n parts and each thread sorts a single part. Then the parts are merged in parallel by pais until a single sorted chunk is obtained. The algorithm performs better if the number of threads is a power of two.

1.2 Implementation

The tests were made by sorting 100 million elements in cases large random integers, small random integers, constant input, increading values and decreasing values.

1.3 Hardware

The computers had the following specifications: Intel Xeon E3-1230v2, 4 cores, 8 thread, 3,3 GHz, RAM: 16 GB, GPU: Nvidia K2000.

1.4 Performance

As expected, performance increased with the number of threads. The multithreded version was in average 3,8 times faster than the single threaded solution for the large random case values. In the other cases there was a slight increase in speed.

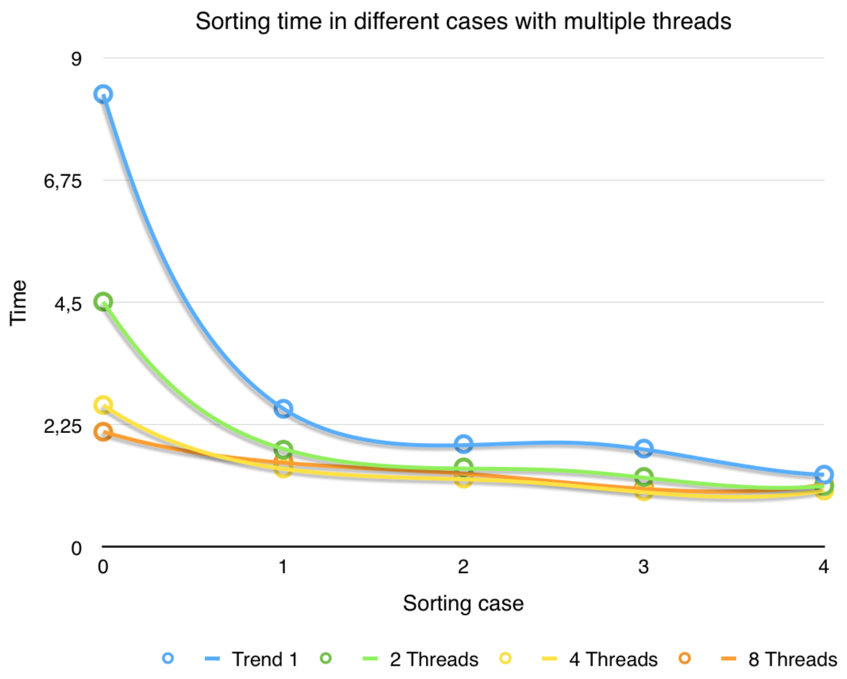

Figure 1: Multithreaded sorting time in cases: 0 = large random elements, 1 = small random elements, 2 = constant input, 3 = increasing values, and 4 = decreasing values.

1.5 CP2 resubmission

NOTE: Resubmission CP2. I updated my cp2 file that was rejected because I misused malloc to free an array memory.

I