Assignment 1 Solutions

Xiaoping Li

February 27, 2014

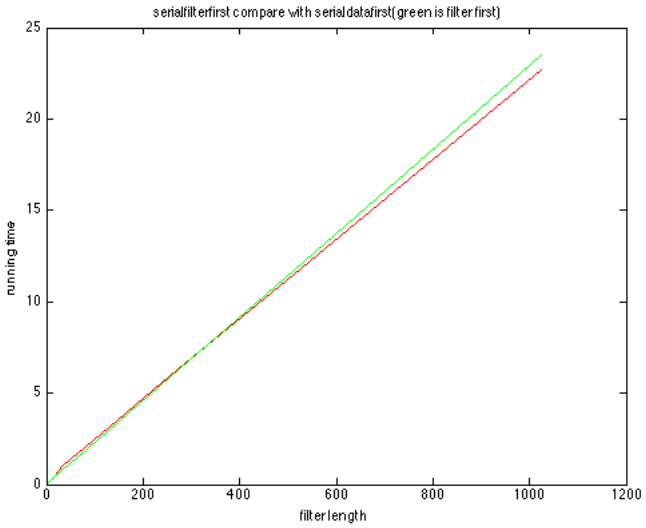
1 Loop Efficiency

Hardware Specifics

I have done my work on AWS a c3.2xlarge instance. All the data has been ran over 20 times. I random pick out 20 and then take an average.

1.1.

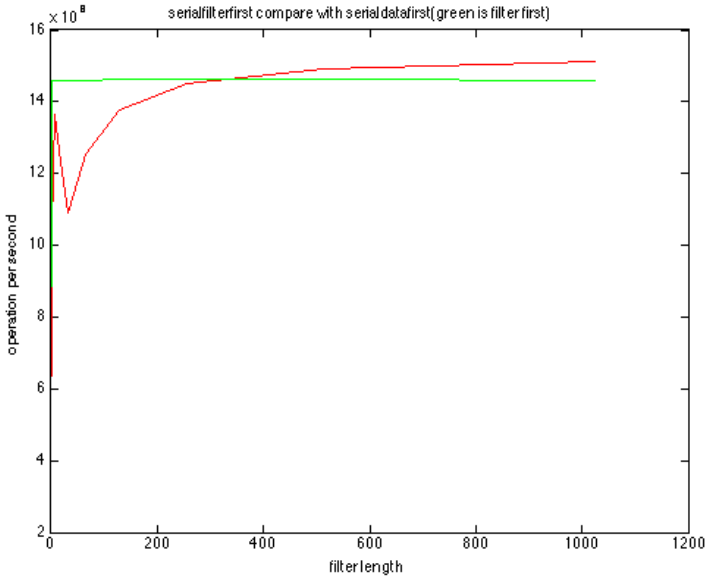
I have generate the "running time vs filter length" in matlab. We can definitelty observe that green line is is a little higher than red line when filter length increase. Green line indicate filter first loop, red line indicates data first loop.



1.2

The normalized figure is listed below. From the figure, we can directly see that with increasing of the filter length, the operation is more utilized. Also, data

first loop has higher operation per second compared with filter first loop.

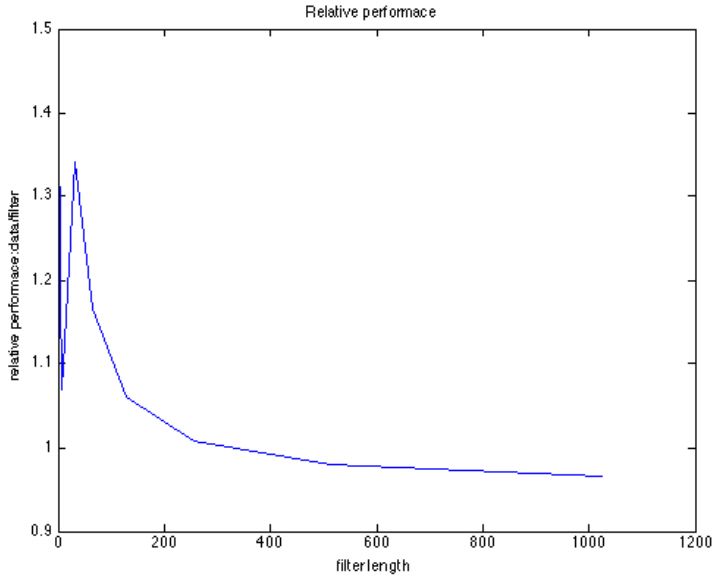


1.3

Compared from figure, we can point out it is faster when data first loop. It is because put the data out of filter loop is better for cache sharing. For example, if we put filter in the outerloop, input_array[x] is updated for each inner loop. Data array is much larger than filter array, so it will cause more burden to the system when data inside of the filter loop.

1.4

The relative performance (data first running time / filter first running time) plot is shown below. With a lower filter length, the data first loop has not much difference with filter first loop. With the increasing of filter length, data first loop becomes faster than filter first. And then reach a constant level.

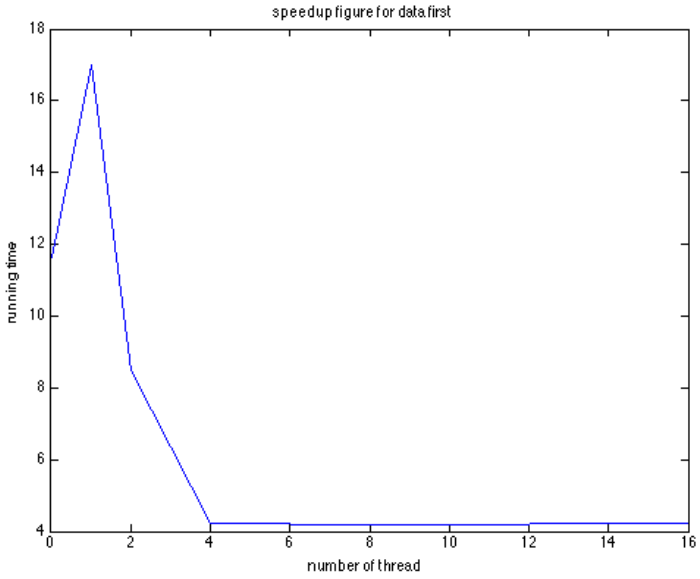


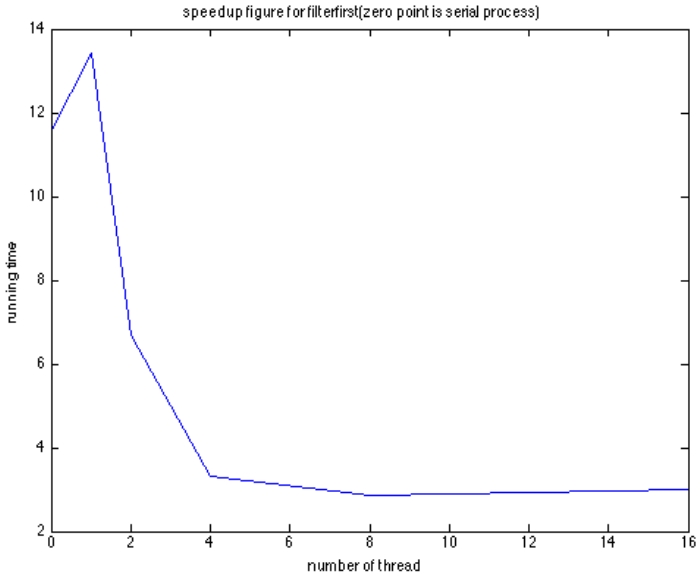
2 Loop Parallelism

speedup and scale up.

2.1

Speed up plot is shown below. I have generated two difference speed up with data first loop and filter first loop.





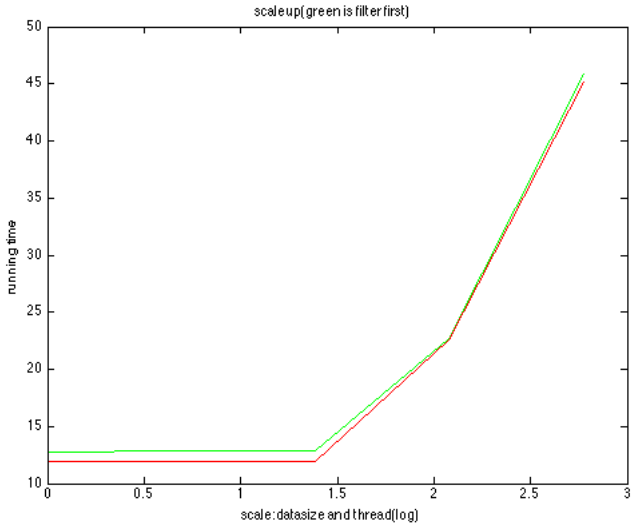
2.2

We can observe that running time is decreased with the increasing of thread until the there are 8 thread. At that point, the running time become constant even increase. The plot tails off around 3 which means 8 thread. The hardware I

am using is on a AWS c3.2xlarge instance with 8 CPUs. Since thread over 8 will cause waiting for other thread which lead to the result. The best performance is related to the hardware. If I am using 16 cores hardware, I belieeve running time will drop down on thread 8.

2.3

The scale up figure is given below. Green line indicate filter first, red line indicates data first.



2.4

From the figure, we can see that it does not keep constant. It remains constant until thread become 8. I am running the simulation on AWS c3.2xlarge with 8 CPUS. So it is its limit on running 8 thread for the better performance.

Parallel Performance.

3.1

Data first parallel code is faster than the filter first parallel code since the former one has a better utilization of the sharing cache which is similar to the serial situation. However, the time difference is not that obvious as serial since running time is spreading over threads when parallel.

3.2

Amdahl's law, if the parallel portion of code is p, number of threads is s, then transform the formula, we can get the linear regression:

$$\frac{1}{n} - 1 = (\frac{1}{s} - 1)p$$

Put data in Matlab, after calculation the estimation of p is 0.8451 and 0.8512 correspond to data first and filter first.

3.3

According to our above results, we can point out that when thread increase, the running time is not decreased with direct ratio. Also in the scaleup figure, we also can see it is not constant with the increasing of data size and thread. In my opinion, I think overhead may be a reason that cause the problem. Since CPU has limits to coordinate different thread based on its hardware. If I am applying hardware with 16 CPU, the performance with higher thread should be better than now.

An Optimized Version

4.1

I have tried several method, the best method is to increase the increment in the for loop which can double the performance. The reason for this case is that every time system arrange a space for the variable which is large for a integer.(only 8 bytes)Code is filterOP.c in the source directory.

4.2

I have tried to collaps the for loop, but it does not make much sense since the performance is even worse than former. The reason is maybe that still we are using the same cache, there is no difference between these two.