Florea Adrian-Paul
Gr. 933

Lab 7

1. Given a sequence of *n* numbers, compute the sums of the first *k* numbers, for each *k* between 1 and *n*. Parallelize the computations, to optimize for low latency on a large number of processors. Use at most 2**n* additions, but no more than 2*log(*n*) additions on each computation path from inputs to an output. Example: if the input sequence is 1 5 2 4, then the output should be 1 6 8 12.

Algorithm:

The main idea of the algorithm used to solve this problem is that each thread increases the sum of the first k elements on the i - k position. If the number of active threads is less than the number of threads, we increase the number of threads working on the resulting array, otherwise, we just increase the partial sum.

Synchronization methods:

A mutex is used to lock individual elements of the resulting array of first k sums. Also, the threads are joined at the end of their execution.

Performance:

Number of elements	Number of Threads	Execution time (ms)
10	5	0.4708
10	10	0.52565
50	10	0.831016
50	25	1.45358
50	50	2.68399