CS3014: SPARSE PARALLEL MULTICHANNEL MULTIKERNEL CONVOLUTION

CONOR MCCAULEY, SEAN ROCHE

REDUCING REPEATED MEMORY ACCESSES

Our first optimisation was to remove the repeated memory accesses to the `kernels` matrix and the `image` matrix.

We then removed the repeated accesses to `output` for each element in the `kernel_starts` interval and replaced them with a local `sum` variable. The appropriate `output` value was incremented by the `sum` after the loop had been exited.

We also removed the initial ‘*zeroing*’ loops as the `output` matrix was already filled with zeroes.

IMPLEMENTING OPENMP

We then configured *OpenMP* with a `#pragma` declaration which allowed us to parallelise the bulk of our code.

We used `collapse(3)` to multithread the image width and height loops as well as the first kernel loop.

We added an `if` condition to the declaration to ensure we only parallelised the code if the number of kernels was greater than or equal to 64 – we chose this number through trial-and-error.

TIMINGS

Input	Average Execution Time
16 16 1 32 32 100	199 µs
64 64 3 256 256 100	124 ms
128 128 3 256 256 100	314 ms
256 256 3 256 256 100	5.01 s