30 March 2018

Make sure you are all up to date with all past assignments by this point

TIME LINE

March 30th (today)

April 2nd

April 9th – **Preliminary code should be done**

April 16th

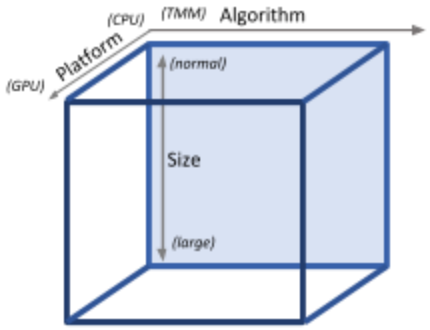
April 23rd – **Final code** must done before this week's meeting

April 30th – for the rest of this time we will be getting preliminary performance data

(Poster fair is probably Wednesday May 2nd (Sanjay will follow up on this))

Problem to be solved: Triangular matrix multiplication (TMM)

Four problems we can be working on in Triangular matrix multiplication



- GPU, CPU, Triangular Matrix Multiplication, and Size

Microbenchmarks - Develop benchmarks that "saturate" various hardware resources

- Latency of Bandwidth to each memory hierarchy (CPU & GPU)
- operation throughput (CPU & GPU)
- bank conflicts (GPU)

modify your code so that one of these resources are saturated

Self organize into 5 potential projects

- microbenchmarks
- 4 axis on cube

Triangular matrix multiplication

- first look for libraries
    - 2 common libraries → BLAS library and MKL (math kernel Library), so far MKL has the fastest version of matrix multiplication

data type

|_D_|_TR__|_MM_|

D – double or S – single precision …

TR – two digits like TR or GE …

MM – identify the function MM is matrix multiplication , SV…

- MM gets two parameters:
    - C = α C + β A*B
    - In Blas – it assumes only one of the matrices is triangular
        - this means that the number of iterations will be ((1/2)*n^3) and therefore n^3 operations
        - so what happens when you multiply two upper right hand triangular matrices ?
            - the number of iterations will be $\sum_{k=I}^{j} A_{ik} * B_{kj}$
            - so in resulting c matrix, the diagonal is when there is only one term left and that is the earliest case when there is a non zero value in the result ,
            - therefore you only have to compute half of the output values
            - so ask yourself how many iterations are there? -> less than ½ n^3
                - this is why we need to develop and optimize our own libraries
        - the MKL delivers a number close to 80% of theoretical machine peak
            - TRMM – this does a bunch of redundant calculations
            - so we can do better

HW FIGURE OUT TOTAL COMPLEXITY of triangular matrix computation
(also think about what happens when you multiply two lower triangle matrices (not hw))
Make sure you have your github set up

we are targeting 2k sized matrices , in normal range for both normal GPU and normal CPU

CUBLAS is a library that implements BLAS operations in CUDA
There is a reference guide available here for BLAS routines. : http://www.netlib.org/blas/blasqr.pdf