Hands-On — CUDA SDK - Libraries, Numerical Accuracy

Siegfried Höfinger

VSC Research Center, TU Wien

March 8, 2020

→ https://tinyurl.com/cuda4dummies/ii/ho4/notes-ho4.pdf

Outline

Hands-On — CUDA SDK - Libraries, Numerical Accuracy

Exercise

Q1) *Given the following matrix,*

$$\mathbf{A} = \begin{pmatrix} 0.30 & -0.61 & 0.40 & 0.37 & -0.49 \\ 0.51 & -0.29 & -0.41 & 0.36 & 0.61 \\ 0.08 & -0.38 & -0.66 & -0.50 & -0.40 \\ 0.00 & -0.45 & 0.46 & -0.62 & 0.46 \\ 0.80 & 0.45 & 0.17 & -0.31 & -0.16 \end{pmatrix}$$

how could we quickly check with the help of CUBLAS that this is actually a matrix consisting of only eigenvectors of a symmetric matrix, M ?

20 min

A1) *Since the eigenvectors of a symmetric matrix M corresponding to different eigenvalues are orthogonal to each other, it follows that the inverse, A^{-1}, is simply the transposed, A^t, making the matrix matrix multiplication, $A^t \times A$ result in the unit matrix, E. Matrix matrix multiplication is a straightforward case for CUBLAS (see below version for download).*

→ https://tinyurl.com/cuda4dummies/ii/t/chck_ev.cu

Exercise

Q2) *Could we make use of CUDA managed unified memory, i.e. cudaMallocManaged(), when calling CUBLAS, for example when modifying the previous case ?*

10 min

A2) *Nope, unfortunately not (see below version for download).*

→ https://tinyurl.com/cuda4dummies/ii/t/chck_ev_v3.cu

Exercise

Q3) *Solve the eigenvalue/eigenvector problem of the following matrix*

$$\mathbf{M} = \begin{pmatrix} 1.96 & -6.49 & -0.47 & -7.20 & -0.65 \\ -6.49 & 3.80 & -6.39 & 1.50 & -6.34 \\ -0.47 & -6.39 & 4.17 & -1.51 & 2.67 \\ -7.20 & 1.50 & -1.51 & 5.70 & 1.80 \\ -0.65 & -6.34 & 2.67 & 1.80 & -7.10 \end{pmatrix}$$

using cusolverDN

20 min

A3) *Eigenvalues are,*
-11.07 -6.23 0.86 8.87 16.09
and eigenvectors those already mentioned in Q1 (see below version for download)

→ https://tinyurl.com/cuda4dummies/ii/t/chck_cusolver_syevd.cu

Exercise

Q4) *Use the 3-step process for NVIDIA Nsight Compute CLI to determine causes of improvement when profiling the series mmm_example_[1-3].cu*

→ https://docs.nvidia.com/nsight-compute/NsightComputeCli/index.html
→ https://tinyurl.com/cuda4dummies/ii/t/mmm_example_1.cu
→ https://tinyurl.com/cuda4dummies/ii/t/mmm_example_2.cu
→ https://tinyurl.com/cuda4dummies/ii/t/mmm_example_3.cu
15 min

A4)

i) *Adding a baseline (using the most simplistic approach) and doing relative comparisons to this baseline is a very straightforward way of identifying improvements in different implementations; for example, mmm_example_2.cu versus mmm_example_1.cu (baseline) immediately points out improvements in terms of memory access, but degradation in SM performance mainly because of fewer instructions per cycle, which is a consequence of a reduced number of warps (eligible as well as active) per cycle making warp cycles per issued instruction increase; also simply looking at Duration [msecond] reveals an almost doubling of the exe-time while memory throughput has dropped from 67 GB/s to 2 GB/s*

ii) *mmm_example_3.cu versus mmm_example_1.cu (baseline) demonstrates a significant reduction in exe-time (Duration) at better SM utilization and comparable memory access (both at 80% of theoretical max, aka SOL); here the inverse trend is observed w.r.2 instructions/cycle and warps dynamics; memory throughput however has also dropped from 67 GB/s to 17 GB/s*