Technische Universität München
Lehrstuhl Informatik V
M. Bader, O. Meister

WS 2013/14
Worksheet 4
December, 11th 2013

Tutorial: HPC - Algorithms and Applications WS 13/14

Complete the following assignments (alone or in a group), and send your source code via e-mail to `meistero@in.tum.de` until Sunday, January, 5th 2014.

Worksheet 4: Heat Equation in CUDA

Assignment 1: cuBLAS + ELLPACK

Write an ELLPACK matrix-vector multiplication kernel for the Heat Equation example code.

a) Open `kernels.cu`, define grid and block size for the call to `ell_mat_vec_mm`

b) Implement the `ell_mat_vec_mm` kernel:

  i) Again, assign one thread to a row and let the kernel perform the inner loop.

  ii) This time, take care that access to the arrays `indices` and `data` is coalesced, access to `x` may be uncoalesced.

  iii) In order to achieve coalesced access, we must imply that entries in `indices` and `data` are in a specific order. Which order is that?

c) Open `poisson.c` and implement `poisson_ellpack`.

  i) Add instructions for cuBLAS context creation and destruction.

  ii) Use cuBLAS in order to execute vector algebra required for simulating the discretized heat equation.

d) Execute the code using `./sparse -H [<matrix size>]`. If you have gnuplot installed you can uncomment `#define GNUPLOT` in `poisson.c` for visual output. Textual output for a matrix of size 64 × 64 should be:

```
t = 0.0000, err = 1.4142e+00, Temperature at x = 0.5: 1.0000e+00
t = 0.6450, err = 9.8181e-03, Temperature at x = 0.5: 6.9696e-01
t = 1.2900, err = 5.1307e-03, Temperature at x = 0.5: 3.8419e-01
t = 1.9350, err = 2.8166e-03, Temperature at x = 0.5: 2.1122e-01
```

```
t = 2.5800, err = 1.5484e-03, Temperature at x = 0.5: 1.1613e-01
(...)
```

Assignment 2: cuSPARSE

Write a cuSPARSE matrix-vector multiplication kernel for the Heat Equation example code.

a) Open `poisson.c` and implement `poisson_cusparse`. A sparse matrix in CSR format already exists.

   i) Add instructions for cuSPARSE context creation and destruction.

   ii) Convert the CSR matrix to HYB format using cuSPARSE

   iii) Call cuSPARSE for matrix-vector multiplication in the time step loop

   iv) Add cuBLAS vector instructions similar to assignment 1

b) Execute the code using `./sparse -H -C [<matrix size>]` and check if the result is consistent with assignment 1d.

c) How do performance of the ELLPACK and the cuSPARSE kernel compare for different matrix sizes?