Assignment 4: Matrix Multiplication in Dynamic Shared Memory

Please read these instructions carefully. Your grade is based upon meeting all requirements stated in this assignment.

Background and Problem Statement

The cuda API provides an environment for the development of host/device paradigm parallel applications which can make use of thousands of synchronous cores. Although subject to inherent communications overhead applications which perform heavy computations and/or conform to a SIMD organization can greatly improve computational runtimes in a GPU environment.

Remember the parallel version of the matrix multiplication problem we discussed in class that used global memory in the GPU. The goal of this assignment is to accelerate that matrix multiplication problem using shared memory constructs.

Program Requirements

- Start with the skeleton program provided (matrix_mul.cu) and then re-architect it as needed to add support for CUDA dynamic shared memory.
- Your final source code MUST be named matrix_mul_shared_dynamic.cu
  - What to do if vim gives you errors about the swap file being present?
  - When your vim crashes without closing things properly, it leaves behind a hidden .swp file. In your case, it is probably named matrix_mul_shared_dynamic.cu.swp.
  - Please note the dot in front of the file name. If you can delete that swp file, you will not get the error.
- For your final submission, you MUST use a value of 18432 for DSIZE.
  - For testing, you can use lower values to make the program run quicker.
- Please find the optimal value of the grid and block size that yields the best performance.
- Report numbers with all the different grid/block sizes you tried.
- Make sure the final files you submit has the best values for grid/block size hardcoded in the code.
  - The grader WILL NOT try out your program with different grid and block sizes.
  - He will run your program and use the time it gives when he runs it to grade your assignment and give you points.
- The top 10 fastest programs will be subjected to a stress benchmark.
  - The top 5 from that pool will receive extra credits.
- You should try to minimize the number of changes needed.
  - Ideally, you only need less than 15 lines of change.
  - We will be using the “wc -l” command to count the number of lines in each program.

Input Files Provided

The following file have been provided to you. All files are available on Owens in the /fs/ess/PAS2661/ /Lab3-Assignment directory.

- Matrix_mul.cu – The skeleton file

- run_script.sh – A script to run the final program and collate the output

Testing Instructions

- Use the NVIDIA C compiler (“nvcc”) only.
  - run_script.sh should already do this.
  - Your program should compile without warnings.
  - Points will be deducted for any warnings
- Logging in to Owens using SSH

  ssh -X -C <username>@owens.osc.edu
- When compiling and making short (less than a couple minutes) test runs, you can use the login nodes.
- For final benchmark runs or when running with the larger value of usleep, please do it on an interactive allocation as described below.
- Getting an interactive allocation

  salloc -N <number_nodes> -A PAS2661 -p gpudebug --gpus-per-node=1 --time=<time_in_minutes>

  e.g. To request for one full GPU compute node with one GPU from the gpudebug partition for 30 minutes, you can use the following command. I am just using 30 as an example. Please use higher values as appropriate.

  -bash-4.2$ -bash-4.2$ salloc -N 1 -A PAS2661 -p gpuserial --gpus-per-node=1 --time=20

  salloc: Pending job allocation 18262768

  salloc: job 18262768 queued and waiting for resources

  salloc: job 18262768 has been allocated resources

  salloc: Granted job allocation 18262768

  salloc: Waiting for resource configuration

  salloc: Nodes o0805 are ready for job <- That is node that was allocated for you

  bash-4.2$ hostname

  owens-login03.hpc.osc.edu <- You're still on the head node when you get the control back

  bash-4.2$ ssh o0805 <- This will move you to the node that was allocated

  <….snip….>

  -bash-4.2$ hostname Now you're on the node that was allocated.

  o0805.ten.osc.edu
- You must load the CUDA module after this manually. The run_script.sh file will do this automatically for you. You must load the “cuda/11.6.1” module.
- -bash-4.2$ module load cuda/11.6.1
- -bash-4.2$ module list
- Currently Loaded Modules:
- 1) xalt/latest 2) gcc-compatibility/8.4.0 3) intel/19.0.5 4) mvapich2/2.3.3 5) modules/sp2020 6) cuda/11.6.1
- Use the nvidia-smi command after loading the CUDA module to make sure that a GPU is present on the system

```
-bash-4.2$ nvidia-smi
Mon Mar 28 09:35:33 2022
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 510.47.03    Driver Version: 510.47.03    CUDA Version: 11.6     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla P100-PCIE...  On   | 00000000:04:00.0 Off |                    0 |
| N/A   26C    P0    25W / 250W |      0MiB / 16384MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

Report Requirements

Provide a 1–2-page report which includes the following (note that accuracy and brevity are prized, needlessly lengthy reports may lower your score).

- You will submit the PDF report on Carmen.
- In your PDF report, you should address the following points
  - Provide a brief explanation describing how you utilized the shared memory.
    - What data did you load into the buffer and why?
    - How did you assign the work to each thread?
  - Were any other additions to the code necessary? Why?
  - Provide a chart or graph which illustrates the scalability (or lack thereof) of your shared memory-base solution and the base non-shared memory solution.
    - Y-axis of the graph will represent time taken for execution of the program as reported by the run_script.sh
    - X-axis should be the block size
    - You should have two lines: one representing the non-shared memory code and one representing your solution.
  - Please choose one grid/block size which gave you the best performance.
  - Hard code this in your sample program so that the grader does not have to pass any additional arguments

- Explain why the grid/block size you chose gave the best performance
- If you saw any unusual or unexpected results, describe them.

Code Submission Instructions

- The run_script.sh file will compile your program and run with different block sizes for both the base solution and your solution.
- On successful completion, the run_script.sh file will create a sub-directory named "cse5441-cuda-lab" and place all necessary files for submission there.
- Submit your cse5441-cuda-lab directory from an Owens login node using the OSC submit system
  - More details about how to submit your assignment from OSC is available at the following website: https://www.osc.edu/resources/getting_started/howto/howto_submit_homework_to_repository_at_osc
- Submit this directory to "Lab3-Assignment"
  - You can use the following command to do the submission
    - "/fs/ess/PAS2661/CSE5441_SPRING24/submit"
  - Note that the system will automatically close at the specified deadline for submission. You will not be able to submit your files after that.
  - Here is a sample set of steps to submit.

./run_script.sh

Program may be using….

<…snip…>

-bash-4.2$ ll cse5441-cuda-lab/

total 24
-rw-r--r-- 1 osc0577 PZS0622 0 Mar 28 11:55 matrix_mul-compilation-output
-rw-r--r-- 1 osc0577 PZS0622 3030 Mar 28 11:55 matrix_mul.cu
-rw-r--r-- 1 osc0577 PZS0622 134 Mar 28 11:55 matrix_mul-output
-rw-r--r-- 1 osc0577 PZS0622 0 Mar 28 11:55 matrix_mul_shared-compilation-output
-rw-r--r-- 1 osc0577 PZS0622 3581 Mar 28 11:55 matrix_mul_shared.cu
-rw-r--r-- 1 osc0577 PZS0622 130 Mar 28 11:56 matrix_mul_shared-output
-rw-r--r-- 1 osc0577 PZS0622 188 Mar 28 11:56 matrix_mul_shared-terminal-output
-rw-r--r-- 1 osc0577 PZS0622 168 Mar 28 11:55 matrix_mul-terminal-output

-bash-4.2$ /fs/ess/PAS2661/CSE5441_SPRING23/submit

****

Hello. This script will submit your assignment assigned by Hari Subramoni to OSC

Note:

Before you run this script, please create one directory which includes all the files you want to submit

The previous submission of the same assignment from you will be replaced by the new submission

Enter the name of assignment and press [ENTER]

>>>Lab3-Assignment

Enter the absolute path of the directory which includes all the files of your assignment lab2 and press [ENTER]

You will submit the current directory if you simply press [ENTER]

>>>/users/PZS0622/osc0577/CSE-5441/CUDA/Homework/cse5441-cuda-lab

Your assignment Lab3-Assignment is submitted successfully. Thank you!

Rubrics

Note to students: I have attempted (to the best of my abilities) to cover all possible cases and create a comprehensive rubric. I am and have been updating these based on feedback/questions I have received over the duration of this course. Even after this, it could still not cover all the questions you may have.

The student should have submitted one program – matrix_mul_shared_dynamic.cu.

1. 70 points for the correct program without any data validation errors
    a. For a correct program, the points will be divided as follows
        i. 50 points for the correctness of the implementation
        ii. 20 points for the quality of the implementation
            1. 10 points will be deducted if the code does not pass a linting test using cpplint
            2. Students are encouraged to use cpplint on their submission code before submitting
            3. See https://github.com/cpplint/cpplint for how to setup and use cpplint
            4. Easiest way to install on Owens

                module load python

                python -m pip install --user cpplint

    b. If you have validation errors, you will lose a minimum of 50 points.
    c. If the program does not run for the grader, he will deduct a minimum of 50 points.
        i. Questions of the type “the program ran fine for me, but not for the grader”, “the program did not hang for me, but it hung for the grader” etc may be considered on a case-by-case basis.
    d. If you forget to submit your program because you did not follow the instructions listed above, you will lose all 70 points that are meant for the program. No excuses.
2. 15 points for getting approximately 5X improvements in performance with a DSIZE of 18432 on the Owens gpuserial partition.
    a. The program should approximately show 5X improvement in compute time as reported by the run_script.sh.
    b. The non-shared memory version of the program will take around 50 seconds to perform the computation. If done well, the shared memory version of the program should take around 10 seconds for the computation.
    c. If the improvement is less than 4X, the student will lose 15 points for the program.

3. 10 points for the report and the clarity with which the student has described the changes
4. 5 points if the program cleans up all resources it allocates
    a. The program should cleanup all resources it allocated.
    b. If it does not – i.e., if there are memory leaks, then the student will lose points based on the percentage of resources they failed to clear up.
5. Extra Credit (10 points)
    a. 5 points if the number of lines of change is less than or equal to 15
    b. 5 points if improvement in compute performance is over 6X
6. Late submissions will lose points
    a. The due date will not be extended
    b. 5 points will be deducted for every 24 hours of delay for a maximum allowable delay of 48 hours.
    c. Beyond 48 hours of delay, the student will receive no points for the assignment.

Academic Integrity

Collaborating or completing the assignment with help from others or as a group is NOT permitted

Copying or reusing previous work done by others is not permitted. You may reuse the work you did as part of this class