Lab1 Test Framework Manual

The test framework can check the correctness and time costs of all your Matrix multiplications in one time of execution.

Downloading

Download the framework with the following command:

$git clone https://github.com/Meso272/CS-211-Lab1.git

Before running

First, complete the Matrix multiply functions in the file **mygemm.c**. Then, compile the codes (on Tardis) with the following commands:
Then enter into the build folder in the framework folder.

$ mkdir build; cd build
$ export CXX=/act/gcc-4.7.2/bin/g++
$ export CC=/act/gcc-4.7.2/bin/gcc
$ cmake ..
$ make
$ chmod +x data.sh

Run the code

Submit the code with SLURM on Tardis with the following command:

$ sbatch submit.sh

Check the output

If the code is error-free, after its execution, you can find the outputs in **data/mydata.txt**. If an error occurred in its execution, you can find error information in a generated file named **lab1.err**.

The file **data/mydata.txt** contains the running result of each Matrix multiplication function. If it is correct, it will show the time cost; If the multiplication result is wrong, it will show the name of the Matrix multiplication function which is wrong.

Some instructions

1. Do not modify any files besides **mygemm.c**, **reg_reuse.c**, **cache_part3.c** and

cache_part.4.c.

2. you only need to submit **mygemm.c** and your PDF report.
3. The framework tests all Matrix multiplication functions in one execution, but the code will not interrupt with empty Matrix multiplication functions (it will only output that the results are wrong). So, you can test and debug one single Matrix multiplication function with leaving others empty.
4. The code file **reg_reuse.c**, corresponds to the coding part of problem 1: register reuse. The matrix sizes in the code are all multiplies of 6 to avoid corner cases, therefore, normally speaking you do not need to modify this file. If you want to deal with corner cases, you can modify the matrix sizes back to 2's exponentials.
5. The code files **cache_part3.c** and **cache_part.4.c** correspond to the part3 and part4 of the problem2: cache reuse. In your experiments, you need to change the value of variable **block_size**. When changing block size, you can modify the value of variable **matrix_dim** to a new value near 2048 which is a multiply of the block size.
6. For the optional bonus questions, test it with the **optimal** function in **mygemm.c** and **cache_part4.c**
7. If there is any question to ask or bug to report with respect to the framework, please contact TA with the email: jliu447@ucr.edu.