CS 211 Project 1

1. Register Reuse

Part #1 & #2.

Dgemm0: For every inner loop(k loop), there will be 3 loads from memory to register and 1 store from register to memory. The statement in the loop has less than 4 float point computation so it can be done in 0.5 cycle. We must repeat this loop for n^3 times. So the final cost of dgemm0 is:

n^3 \* 400.5

If n=1000, the cost is

4.005 \* 10^11

Given the clock frequency is 2GHz, the time is 200.25 seconds. It wastes 400/400.5 of total time to access operands from memory, which is 200 seconds.

Dgemm1: n^2 times for load and store c. n^3 times for load a and b and compute a\*b. Thus, the cost can be presented by:

n^2 \* 200 + n^3 \* 200.5

Given the n=1000 and frequency=2GHz, the time is 100.35s. The time spent on accessing operands from memory is 100.1s.

Execution Time & Performance:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| n | Time (seconds) | | | Performance (GFLOPS) | | |
| dgemm0 | dgemm1 | dgemm2 | dgemm0 | dgemm1 | dgemm2 |
| 64 | 0.007182 | 0.004872 | 0.001923 | 0.07300028 | 0.10761248 | 0.27264067 |
| 128 | 0.039573 | 0.025783 | 0.008413 | 0.10598903 | 0.16267711 | 0.49855034 |
| 256 | 0.283726 | 0.156723 | 0.051325 | 0.11826351 | 0.21410024 | 0.6537639 |
| 512 | 3.127462 | 1.893728 | 0.731275 | 0.08583172 | 0.14174974 | 0.36707867 |
| 1024 | 25.557463 | 17.926471 | 7.384723 | 0.0840257 | 0.119794 | 0.29080084 |
| 2048 | 443.911263 | 265.426373 | 99.790432 | 0.03870113 | 0.06472555 | 0.17215948 |

Correctness is verified using maximum difference.

How To Run

Make 1-p1

Sbatch 1.p1p2.job.sh

Part #3

Use 3x3 scheme to increase the utilization of registers. However, if we allocate three 3x3 matrix in register, the total number will be more than 16. So I put 3 elements of matrix a and b each step. As a result, the total register used is 15. I scale up the entire matrix dimensions to avoid boundary condition problem.

Execution Time:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n | Time (seconds) | | | |
| dgemm0 | dgemm1 | dgemm2 | dgemm3 |
| 66 | 0.003262 | 0.00199 | 0.000843 | 0.000742 |
| 132 | 0.029175 | 0.017559 | 0.007097 | 0.005874 |
| 258 | 0.263643 | 0.147071 | 0.062493 | 0.04656 |
| 516 | 2.413283 | 1.43728 | 0.625699 | 0.423897 |
| 1026 | 19.001283 | 12.314581 | 5.241118 | 2.857817 |
| 2052 | 192.390376 | 107.058477 | 65.766212 | 30.809667 |

Performance:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n | Performance (GFLOPS) | | | |
| dgemm0 | dgemm1 | dgemm2 | dgemm3 |
| 66 | 0.17626977 | 0.2889407 | 0.68207829 | 0.77492183 |
| 132 | 0.15766704 | 0.26197027 | 0.64815218 | 0.78310112 |
| 258 | 0.13027854 | 0.23354043 | 0.54961394 | 0.73769381 |
| 516 | 0.11385991 | 0.19117791 | 0.43915076 | 0.64821452 |
| 1026 | 0.11368133 | 0.17540923 | 0.4121432 | 0.75585356 |
| 2052 | 0.08982117 | 0.16141393 | 0.26275999 | 0.56088659 |

Correctness is verified using maximum difference.

How To Run

Make 1-p3

Sbatch 1.p3.job.sh

2. Cache Reuse

Part 1

When n=10000

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Cache Miss Per Element | | | Number of Cache Read | | | Miss Rate |
| A | B | C | A | B | C |
| ijk/jik | n for a[,k] | k%10==0 | n | 1 | n^3 | n^3 | n^2 | about 55% |
| ikj/kij | 1 | n for b[,j] | j%10==0 | n for c[,j] | j%10==0 | n^2 | n^3 | n^3 | about 10% |
| jki/kji | n | 1 | n | n^3 | n^2 | n^3 | 100% |

When n=10

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Cache Miss Per Element | | | Number of Cache Read | | | Miss Rate |
| A | B | C | A | B | C |
| ijk/jik | 1 if k%10==0 | 1 if k%10==0 | 1 if k%10==0 | n^3 | n^3 | n^2 | 1.43% |
| ikj/kij | 1 if j%10==0 | 1 if j%10==0 | 1 if j%10==0 | n^2 | n^3 | n^3 |
| jki/kji | 1 if i%10==0 | 1 if i%10==0 | 1 if i%10==0 | n^3 | n^2 | n^3 |

Part 2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Cache Miss Per Element | | | Number of Cache Read | | | Miss Rate |
| A | B | C | A | B | C |
| ijk/jik | n/10 if k%10==0 | n/10 if k%10==0 | 1 if k%10==0 | n^3 | n^3 | n^2 | About 0.95% |
| ikj/kij | 1 if j%10==0 | n/10 if j%10==0 | n/10 if j%10==0 | n^2 | n^3 | n^3 |
| jki/kji | n/10 if i%10==0 | 1 if i%10==0 | n/10 if i%10==0 | n^3 | n^2 | n^3 |

Part 3