Lab 5 report

matmul.cpp before and after optimization were ran on both arm system server(**`arm.csc.calpoly.edu`**) and intel system server(**`127x01.csc.calpoly.edu`**) to compare the performance. In overall, intel system are much faster than arm system. But the code optimization on intel server is not as much as effective on arm server.

For the original code (row-major order and without loop unrolling), the runtime on Arm is 267.88 seconds while on Intel server is 59.32 seconds. Obviously, Intel server is much faster. They have almost the same cache misses. Intel system have less cache references than arm system.

After first step optimization (just loop unrolling): the run time on arm system has significant improvement which reduced to 103.68 s. Cache misses was reduced 4 times and cache reference was reduced two times. While on intel system, the runtime slightly reduced to 54.05 s, and have no improvement on cache misses and cache references.

For the second step optimization (just column-major order): the run time on arm system also has significant improvement compared with original code which reduced to 97.99 s. Cache misses was exponentially reduced but cache reference has no improvement. While on intel system, the runtime was also significantly reduced 40.87 s, still have no improvement on cache misses and cache reference.

For the third step optimization (loop unrolling and column-major order): the run time on arm system also dramatically improvemed which reduced to 61.20 s. Cache misses was exponentially reduced but cache reference reduced by two times compared with original code. While on intel system, the runtime was also significantly reduced 34.30 s, still have no improvement on cache misses and cache reference.

In conclusion, the code optimization can dramatically improve the performance on arm server on run time, cache misses and cache reference. It can also help to improve certain run time on intel server but has no much help on improving the cache misses and cache reference.

The Chart table for the data collected and terminal output are as below.

Chart:

matmul	`arm.csc.calpoly.edu`	`Any Intel system 127x01.csc.calpoly.edu`
Average CPI	6.6667	2.2727
Instructions	80,171,881,631	113,795,805,214
Branch-misses	86,188,033	13,090,759
Cache Misses	8,631,558,601	8,714,163,803
Cache References	21,007,994,480	14,746,538,660
Runtime (Measured)	267.88 seconds	59.32 seconds

On ARM Server	**Just Loop Unrolling**	**Just Column-Major**	**Loop Unrolling plus Column-Major**
Runtime	**103.68 s**	**97.99 s**	**61.20 s**
Cache Misses	**2,169,702,700**	**269,312,525**	**138,572,020**
Cache References	**12,418,059,883**	**21,007,994,469**	**12,418,059,879**

On Intel Server	**Just Loop Unrolling**	**Just Column-Major**	**Loop Unrolling plus Column-Major**
Runtime	**54.05 s**	**40.87 s**	**34.30 s**
Cache Misses	**8,667,092,099**	**8,697,386,240**	**8,693,601,846**
Cache References	**14,581,297,602**	**18,982,462,679**	**18,460,622,353**

Step 1:

```
zliang15@arm:~/315/pipelines-and-cache-woshitiancai42 $ make perf1
perf stat -e cycles,instructions,branch-misses,cache-misses,cache-references ./mm > myPe
rf.txt
Printing Result Matrix:
**************************************************
Done.

 Performance counter stats for './mm':

   535,023,753,103      cycles:u
    80,171,881,631      instructions:u            #    0.15  insn per cycle
        86,188,033      branch-misses:u
     8,631,558,601      cache-misses:u            #   41.087 % of all cache refs
    21,007,994,480      cache-references:u

     267.880883011 seconds time elapsed

     267.783020000 seconds user
       0.029998000 seconds sys
```

```
zliang15@127x01:~/315/pipelines-and-cache-woshitiancai42 $ make perf1
perf stat -e cycles,instructions,branch-misses,cache-misses,cache-references ./mm > myPerf.txt
Printing Result Matrix:
**************************************************
Done.

 Performance counter stats for './mm':

   255,920,099,240      cycles
   113,795,805,214      instructions              #    0.44  insn per cycle
        13,090,759      branch-misses
     8,714,163,803      cache-misses              #   59.093 % of all cache refs
    14,746,538,660      cache-references

      59.317970716 seconds time elapsed

      58.979319000 seconds user
       0.015988000 seconds sys
```

Step 2，3: Loop Unrolling Column-major order

On ARM server

1. Just loop unrolling

```
zliang15@arm:~/315/pipelines-and-cache-woshitiancai42 $ make perf
perf stat -e cycles,instructions,branch-misses,cache-misses,cache-references ./mm > myou
t
Printing Result Matrix:
**************************************************
Done.

 Performance counter stats for './mm':

   206,330,961,108      cycles:u
    54,343,631,227      instructions:u            #    0.26  insn per cycle
        82,076,220      branch-misses:u
     2,169,702,700      cache-misses:u            #   17.472 % of all cache refs
    12,418,059,883      cache-references:u

     103.678550774 seconds time elapsed

     103.288717000 seconds user
       0.049994000 seconds sys


zliang15@arm:~/315/pipelines-and-cache-woshitiancai42 $ make compare
./compare.py myout output.gold
All values within both files are within a tolerance of 1.
```

2. Just Column-Major

```
zliang15@arm:~/315/pipelines-and-cache-woshitiancai42 $ make perf
perf stat -e cycles,instructions,branch-misses,cache-misses,cache-references ./mm > myou
t
Printing Result Matrix:
**************************************************
Done.

 Performance counter stats for './mm':

   195,202,705,156      cycles:u
    88,717,013,861      instructions:u            #    0.45  insn per cycle
        82,748,360      branch-misses:u
       269,312,525      cache-misses:u            #    1.282 % of all cache refs
    21,007,994,469      cache-references:u

      97.992677649 seconds time elapsed

      97.715149000 seconds user
       0.000000000 seconds sys


zliang15@arm:~/315/pipelines-and-cache-woshitiancai42 $ make compare
./compare.py myout output.gold
All values within both files are within a tolerance of 1.
```

3. Loop unrolling and column-major

```
zliang15@arm:~/315/pipelines-and-cache-woshitiancai42 $ make perf
perf stat -e cycles,instructions,branch-misses,cache-misses,cache-references ./mm > myou
t
Printing Result Matrix:
**************************************************
Done.

 Performance counter stats for './mm':

   121,654,861,293      cycles:u
    56,492,482,410      instructions:u            #    0.46  insn per cycle
        79,786,419      branch-misses:u
       138,572,020      cache-misses:u            #    1.116 % of all cache refs
    12,418,059,879      cache-references:u

      61.200454469 seconds time elapsed

      60.879192000 seconds user
       0.029994000 seconds sys


zliang15@arm:~/315/pipelines-and-cache-woshitiancai42 $ make compare
./compare.py myout output.gold
All values within both files are within a tolerance of 1.
```

On Intel server:

1. Just loop unrolling

```
zliang15@127x01:~/315/pipelines-and-cache-woshitiancai42 $ make perf
perf stat -e cycles,instructions,branch-misses,cache-misses,cache-references ./m
m > myout
Printing Result Matrix:
**************************************************
Done.

 Performance counter stats for './mm':

   233,135,172,861      cycles

   113,788,556,187      instructions              #    0.49  insn per cycle

        13,081,735      branch-misses

     8,667,092,099      cache-misses              #   59.440 % of all cache refs

    14,581,297,602      cache-references


      54.046190755 seconds time elapsed

      53.711558000 seconds user
       0.031968000 seconds sys
```

2. Just Column-Major

```
zliang15@127x01:~/315/pipelines-and-cache-woshitiancai42 $ make perf
perf stat -e cycles,instructions,branch-misses,cache-misses,cache-references ./m
m > myout
Printing Result Matrix:
**************************************************
Done.

 Performance counter stats for './mm':

   175,475,978,350      cycles

   208,312,488,276      instructions              #    1.19  insn per cycle

        17,257,385      branch-misses

     8,697,386,240      cache-misses              #   45.818 % of all cache refs

    18,982,462,679      cache-references


      40.870658659 seconds time elapsed

      40.527485000 seconds user
       0.019980000 seconds sys
```

3. Loop unrolling and column-major

```
zliang15@127x01:~/315/pipelines-and-cache-woshitiancai42 $ make perf
perf stat -e cycles,instructions,branch-misses,cache-misses,cache-references ./m
m > myout
Printing Result Matrix:
**************************************************
Done.

 Performance counter stats for './mm':

   147,183,899,047      cycles

   165,329,897,241      instructions              #    1.12  insn per cycle

        14,110,529      branch-misses

     8,693,601,846      cache-misses              #   47.093 % of all cache refs

    18,460,622,353      cache-references


      34.303578201 seconds time elapsed

      33.986229000 seconds user
       0.023976000 seconds sys
```