Different ways of allocating arrays

1. int a[m*n] and int a[m][n] will cause `chkstk.asm` throws errors for memory conflict for matrix which bigger than 256*256.
2. int *a is the best way to do this one, has a whole block of memory for each matrix and easy to access. Only down side is the code seems not that clean with signs at the left side of equations.
3. int **a will cause additional work on free memory and communicate between methods by pointer, but also work if really needed.

Experiment environment

- the environment is Intel(R) Xeon(R) CPU E5-2699 v4 @ 2.20GHz with Gcc (Ubuntu 4.9.3-13ubuntu2) 4.9.3. The original version of the code is built under windows 10 V1703 on visual studio 15.3.4 with intel core i7-7700 @4.20GHz, working well in both environment. All data are from Hydra2 server.
- The floating with -O3 data sets are being showed in Figure 1. And all data for interesting findings are being showed in Figure 2.
- The random range increased a significant amount when data class switch from int to float.
- The optimizer list is "-O3 -funroll-loops -march=native -mfpmath=sse -ftree-loop-distribution"

Interesting findings

- Optimize always reduce the runtime of the program no matter the size and class of the data
- The special one which using my own optimizer list, has a very unstable performance during the process. Overall it seems working well on integer data, but for float data it often increases the runtime and sometimes reduced a small amount that can be ignored.
- The data size increase from 2048 to 4096 increased by 4 times, but the runtime only increased by 8 times. Even the triple loop should cause much more of increase by having a n^3 running time.
- The runtime increase amount by having float data instead of integer data is only less than 10% before 2048, which is much lesser than expected. And after 2048 the runtime is reduced a little bit.
- The runtime reduce amount by any optimize always between 55% to 75%, which means the optimize always works well against any kind of matrix multiply problem.
- The memory access optimize is so important that reduced the runtime of the matrix size 4096 set from 1500+ secs to 280 secs with no optimizer.
- The special settings set sometimes has higher runtime than the O3 set.
- Change the data type from float to double has only a small amount effect, and sometimes double has longer runtime than float.

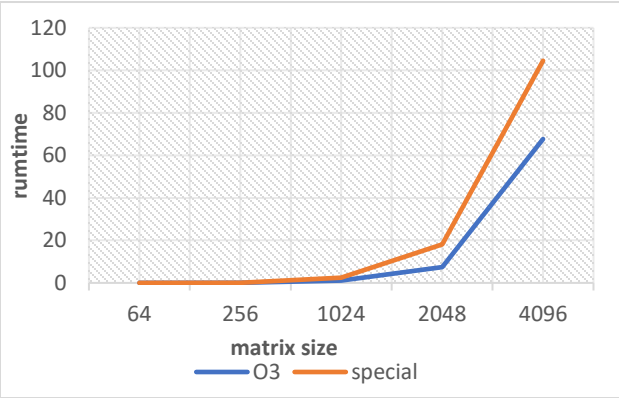

Figure 1 Float with O3 and special

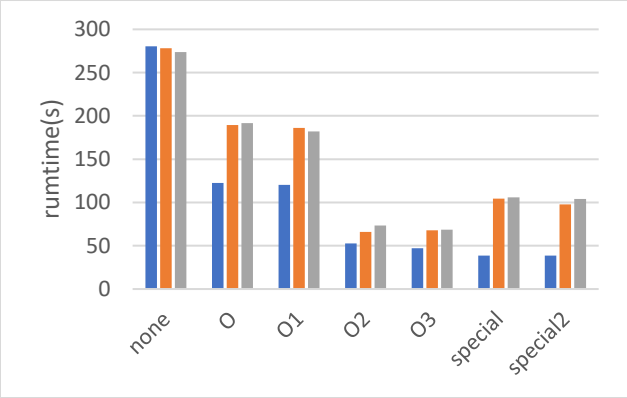

Figure 2 All 15 data sets for 4096 size

Score (for the instructor): _____