By Jiahui Tang

Reported Optimized Result：

(1) GCC -Ofast:
    (1) Best elapsed time: 2.50645 s
    (2) Best real time: 0.195s
(2) ICC -fast:
    (1) Best elapsed time: 0.75382 s
    (2) Best real time: 0.129s

Best improvements:
(1) in terms of elapsed time: 24.606x faster
(2) In terms of real time: 144.193x faster

No optimization:

```
ubuntu@ip-172-31-77-30:~$ gcc -DUSE_CLOCK slowpoke.c -o slowpoke -lm
ubuntu@ip-172-31-77-30:~$ time ./slowpoke
*****************************************************
||c||_F = 2630354827429312.50
Elapsed Time: 18.5493 s.

real    0m18.601s
user    0m18.569s
sys     0m0.032s
ubuntu@ip-172-31-77-30:~$
```

— —
Details:

Version 1 - GCC with slightly lower result marginal error but executed slower

1. Launch m5.2xlarge instance
2. To ensure that it doesn't attempt to sync j and k between threads, we could set them to private versions. So I also moved declaration into the loop body to parallelize them.
3. Change matrix b's index from b[k][j] to b[j][k] to improve memory locality;

Since matrix b[i][j]= I*j thus it is a symmetric matrix, so we can directly switch j and k;
3. 4* unrolling the inner loop

```
get_time(&tstart);
# pragma omp parallel for
for (i=0; i<N; i++)
  {
  for(int jj=0; jj<N; jj++)
    for (int kk=0; kk<N; kk+=4) {
      c[i][jj] += a[i][kk] * b[jj][kk] +  a[i][kk+1] * b[jj][kk+1] +  a[i][kk+2] * b[jj][kk+2] +  a[i][kk+3] * b[jj][kk+3];
    }
  }
  get_time(&tend);
```

4. add OpenMP tags as indicated #pragma omp parallel for

3. set thread = 16;

3. Compile using
~~##gcc -fopenmp -Ofast -march=native -DCLOCK_REALTIME seq_mm.c timing.c -o omp_seq~~

gcc -fopenmp -Ofast -march=native -DUSE_CLOCK seq_mm.c -o omp_seq

Result

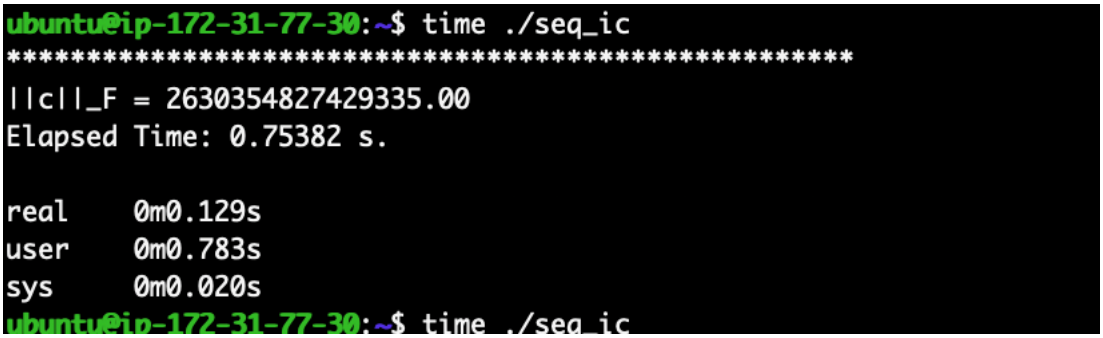
```
ubuntu@ip-172-31-77-30:~$ time ./omp_seq
**************************************************
||c||_F = 2630354827429328.00
Elapsed Time: 2.50645 s.

real 0m0.195s
user 0m2.524s
sys  0m0.079s
```

real time reported with screenshot:

— —

Version 2 (ICC- faster but with slightly more result marginal error)

1. Add apt repo via https://software.intel.com/content/www/us/en/develop/articles/installing-intel-oneapi-toolkits-via-apt.html
2. Run
   apt-get install intel-oneapi-compiler-dpcpp-cpp-and-cpp-classic
3. Run
   source /opt/intel/oneapi/setvars.sh
4. Compile by:
   icc -fopenmp -fast -DUSE_CLOCK ./seq_mm.c -o seq_ic
5. Screenshots:



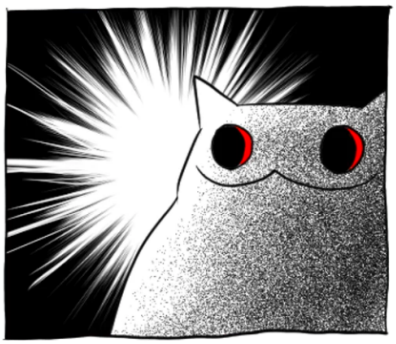
```
ubuntu@ip-172-31-77-30:~$ OMP_NUM_THREADS=16
ubuntu@ip-172-31-77-30:~$ time ./seq_ic
**************************************************
||c||_F = 2630354827429335.00
Elapsed Time: 1.48407 s.

real    0m0.129s
user    0m1.512s
sys     0m0.036s
ubuntu@ip-172-31-77-30:~$
```

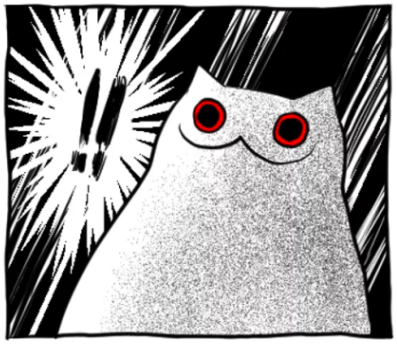
Attachements:

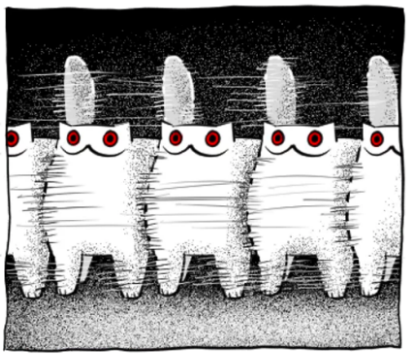
8 threads:
ubuntu@ip-172-31-77-30:~$ OMP_NUM_THREADS=8 time ./seq_ic
****
||c||_F = 2630354827429335.00
Elapsed Time: **0.962459 s.**
0.98user 0.02system **0:00.15elapsed** 649%CPU (0avgtext+0avgdata 55504maxresident)k
0inputs+0outputs (0major+13461minor)pagefaults 0swaps

16 threads:
ubuntu@ip-172-31-77-30:~$ OMP_NUM_THREADS=16 time ./seq_ic
****
||c||_F = 2630354827429335.00
Elapsed Time: **1.48194 s.**
1.52user 0.02system **0:00.12elapsed** 1207%CPU (0avgtext+0avgdata 55856maxresident)k
0inputs+0outputs (0major+13549minor)pagefaults 0swaps

Meme:

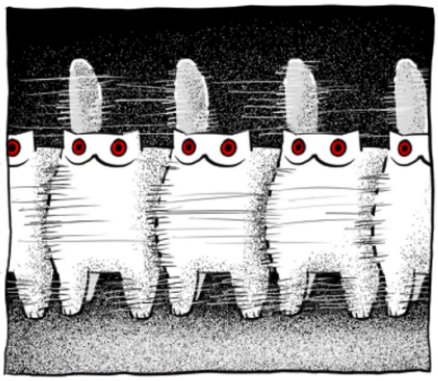




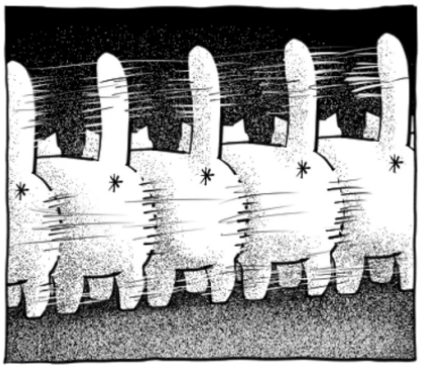


4 X unrolling!

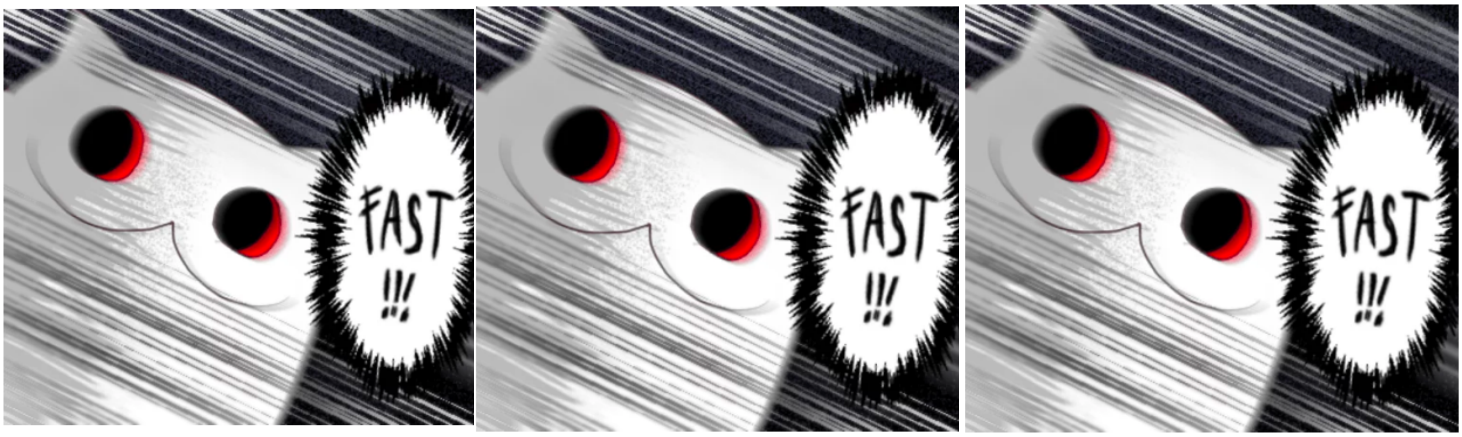
Try more unrolling!



Oops, No, back to
4 X unrolling!



Turn around, switch j
and k index!



Meme Reference:
Meawbin
http://meawbinneko.com/meawbincomic.html