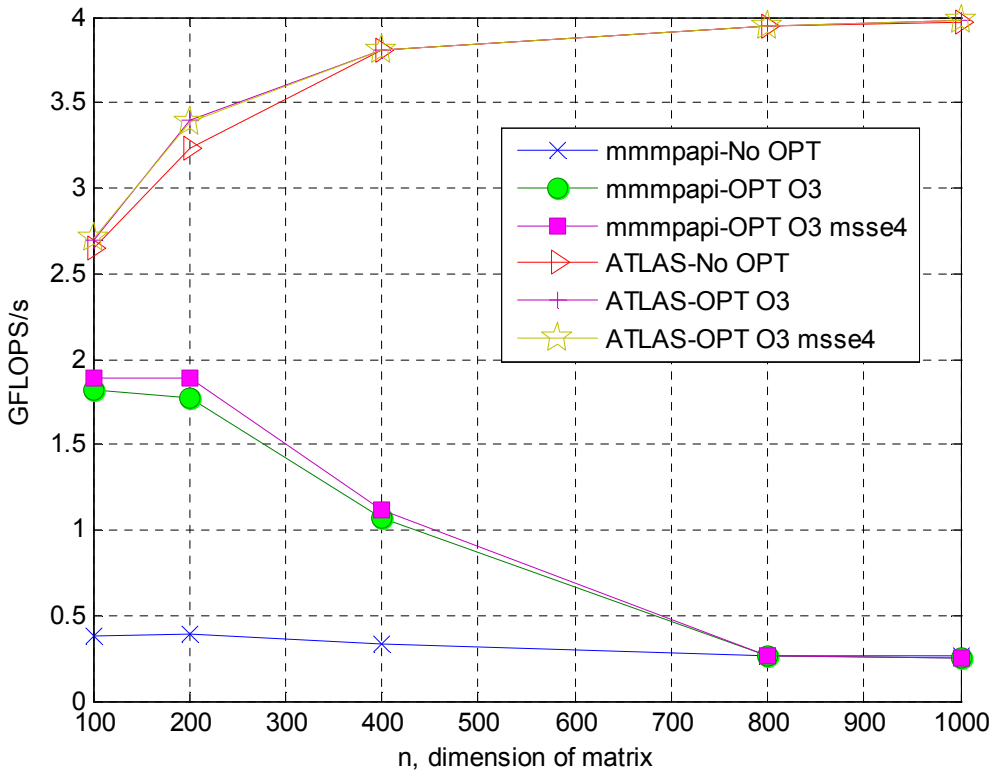
Report for HW05:

Part 1. Because PAPI_get_virt_usec() would give the actual time of running our process rather than real clock time which might include other applications running time which are not the exact time of our application execution.

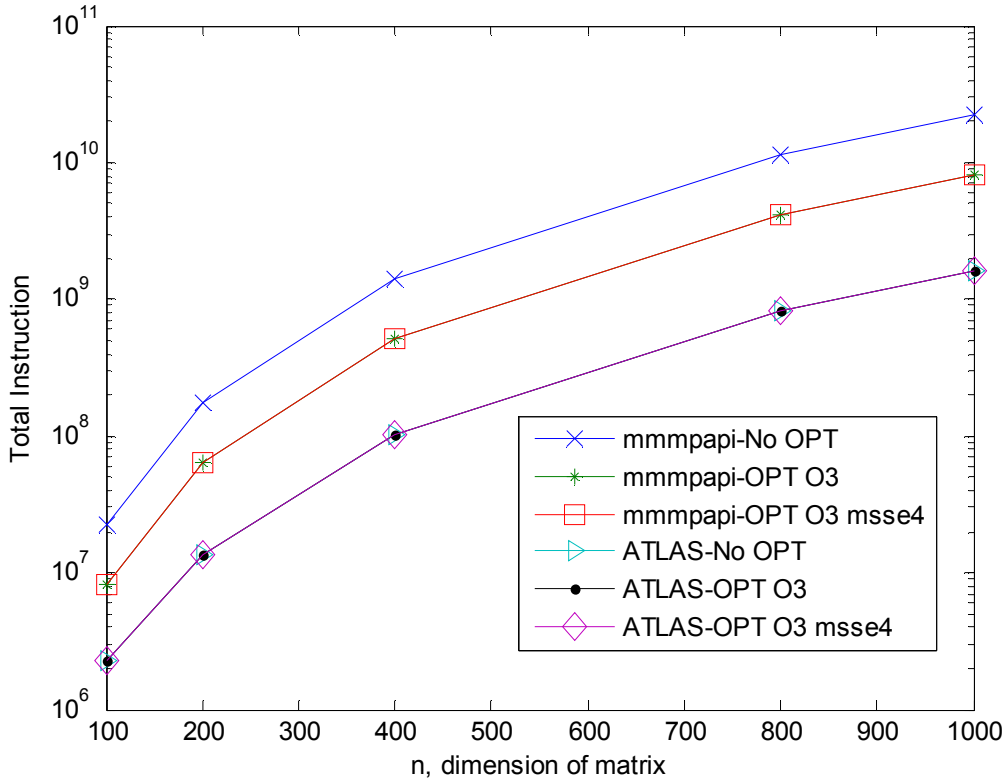
Part 2.



As it is obvious in this figure, for mmmpapi the change of compiler optimization from NO optimization to O3 significantly improves the behavior in dimensions less than 800. The results of O3 and O3 msse4 are very close to each other.

However, the behavior of ATLAS curve is completely different, i.e. GFLOPS/s increases with increasing dimension while in the other method GFLOPS/s decreases with increasing matrix dimension. Generally, the optimization of compiler cannot enhance the performance to be as good as that of ATLAS.

Part 3.



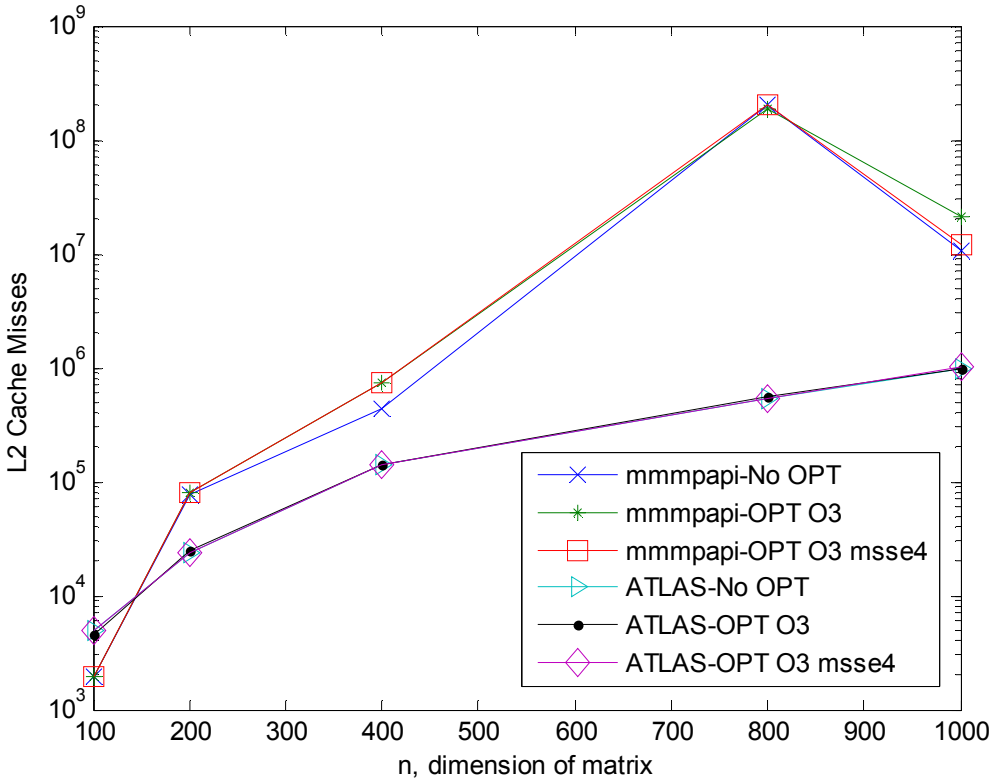
Compiler optimization does not change the results of ATLAS. However, the change from No to O3 optimization decreases the number of instructions of mmmpapi significantly.

It seems that decrease of the number of instruction and enhancement of the performance are happening with the generally similar trend, but with some exceptions. In other words, it could be said that at least performance is a strong function of number of instruction. The exceptions:

1. The difference between number of instructions of NO optimized and O3 optimized mmmpapi remains unchanged while increasing dimension. while the change in performance from NO to O3 is a decaying function of matrix dimension. This is also true for ATLAS.

2.There seems no difference between instructions of different level optimized ATLAS. However, the performance of high level optimized ATLAS is slightly better at low matrix dimension.

Part 4.



One conclusion of this graph is the fact that optimization does not have a significant effect on L2 Cache Misses. One surprising thing is falling of Cache misses in going from 800 to 1000 dimension. I expected rise rather than decrease.

This graph tells me that the naive way of matrix multiplication surely has memory bound problem. However, what I guess is the fact that this L2 cache misses affect the efficiency of CPU usage. So instead of getting the maximum capability of CPU, we are not doing a part of our possible operations. In another words, I think that the efficiency of using CPU and L2 cache is interrelated.

We have learned two methods of optimization. Loop unrolling and Blocking. I think Blocking method could be more effective and sophisticated and I guess that ATLAS is using that. However, it might also use the methods that is used by compiler itself as optimization. It could be something like loop unrolling but in another similar manner.

So, when blocks are read to cache in each loop, then the arguments of A, B and C which are necessary for multiplication are present in Cache and therefore this method decreases the amount of cache misses.

Part. 5

I think L1 and L3 cache misses are also important. because, the access to them are less expensive than going to memory itself in each call.

Also, Floating Point Operations are a good way to know the exact operations that are taking place.