Project 1B: Cache Simulation

This is my second attempt at writing this reflection and I hope this one is comprehensive. In Project 1B, we had to make observations on the usage of cache by our sorting algorithms using various combinations of the block size, number of blocks and the associativity of cache. For this, we were instructed to use the “Cache Simulator” tool available in PLP tool.

This tool is really wonderful as we can actually view how much of our code was cached in the tool’s output. All we need to do is mention the number of blocks, block size and associativity of cache and this will be the definition of our cache. In addition to this, we can mention whether we want to cache the data alone, or the instructions alone or both. Also, we can choose between write-through(s) and write-back(s) caching. It is really this simple to setup a cache and view the caching mechanism.

Thanks to the tool, it was easy for me to understand how the caching changes with varying number of blocks, block sizes and the associativity. I chose to cache only data and not the instructions in order to clearly observe data caching.

One observation which deviated from my expectations was that the read-hits did not really change when associativity was changed.

One major issue that my team faced was specific to the insertion sort algorithm. We were not getting any cache related information for this logic. On exploring further, we realized that the code was not exited properly and hence cache usage could not be tracked for that piece of code. After the presentation of Project 1B, I re-did the code to make sure that I had a working copy with me.

Working on this project helped me in visualizing the cache usage and its change based on the changing block sizes, number of blocks and associativity. The clarity I now have on the concept of caching cannot be attained without using the simulation tool.

Going further, I will write my code by keeping in mind the way caching happens so that its execution is very efficient.