A white background with black text

Description automatically generated

1. Cache Size

If the cache size is too small, it risks not being able to hold frequently accessed data or instructions. I.e., the miss rate increases, and the processor will have to fetch data from lower and slower levels of cache and memory more often.

As the cache size increases, relative access time generally increases. I.e., the time it takes to search the cache for a specific piece of data increases, the hit time. Additionally, there’s the concept of diminishing returns due to the principle of locality since most programs tend to hit smaller, localized regions of space at a time rather than uniformly.

1. Line Size

If the line size is too small, more blocks are needed to fill the cache which creates a lot of overhead since the processor will keep needing to go to lower, slower levels of cache and memory to fetch the needed data. Additionally, for programs that need to continuously access a localized region of space, it might miss adjacent data that’ll need to be accessed which further reduces performance and efficiency.

If the line size is too big, you risk wasting space and bandwidth since you’re prefetching data that may not be needed.

1. Associativity

If associativity is too small, it leads to high miss rates if multiple blocks of memory frequently accessed map to the same cache block since you’re constantly having to switch them out.

On the other hand, if the associativity is too high, you may have to search the entire cache for a block, making it slower.

A screenshot of a computer code

Description automatically generated

Compulsory misses will only occur at and since the cache is initially empty. The problem statement is set up conveniently to allow us to pull the entirety of matrix A and B in. We can do that at the start so, 2 misses.

Since the L1 cache is “small”, each iteration causes new data to be pulled in and successive accesses to and are going to compete with existing cache sets. There are iterations, so 16384 total iterations. Since we are pulling from two sets, there are load operations. Due to the finite space of our cache, there is going to be conflict misses.

A white background with black text

Description automatically generated

