Computer Organization Lab-7

Members

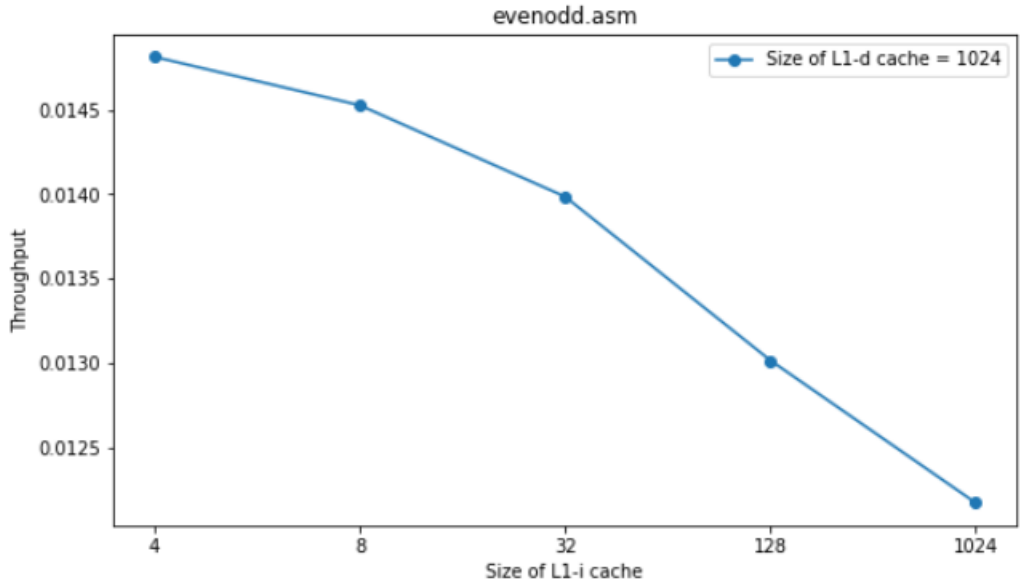
Aiswarya H, 111901006

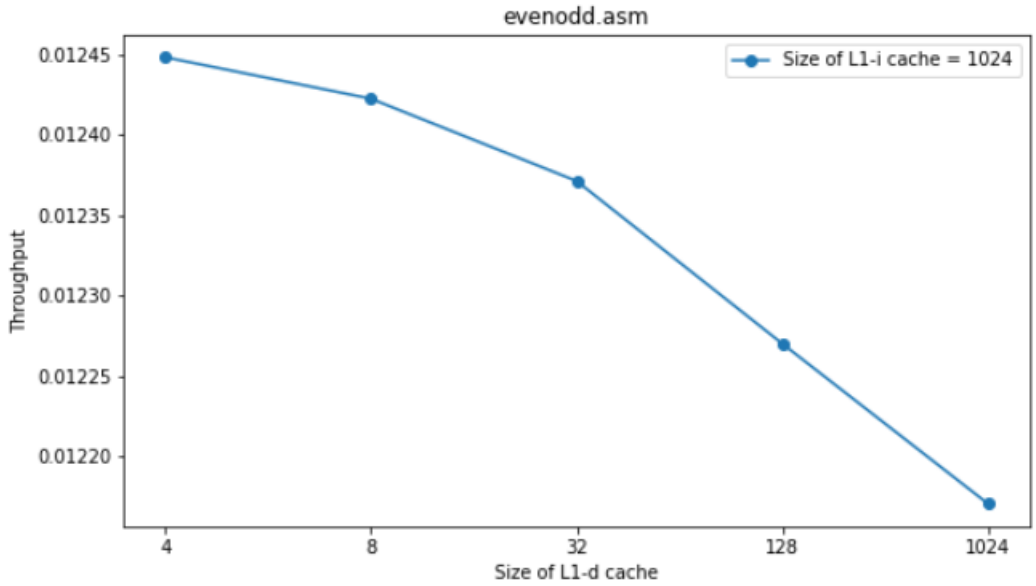
Joel Sam Mathew, 111901026

Reports

In our implementation of Cache.java, when a compulsory miss occurs, we store only the corresponding data in the cache (not including neighbours).

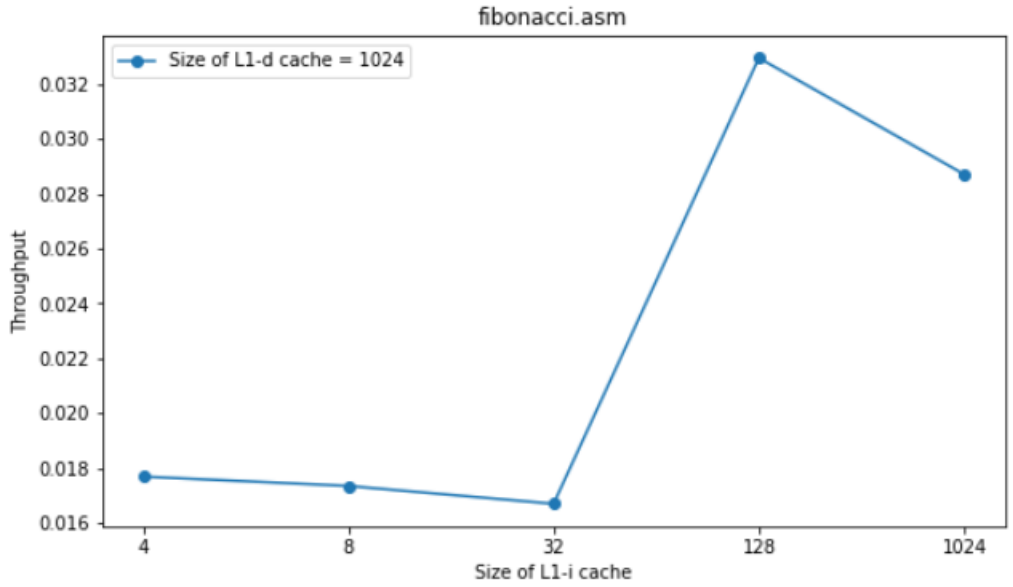
1. even-odd.asm

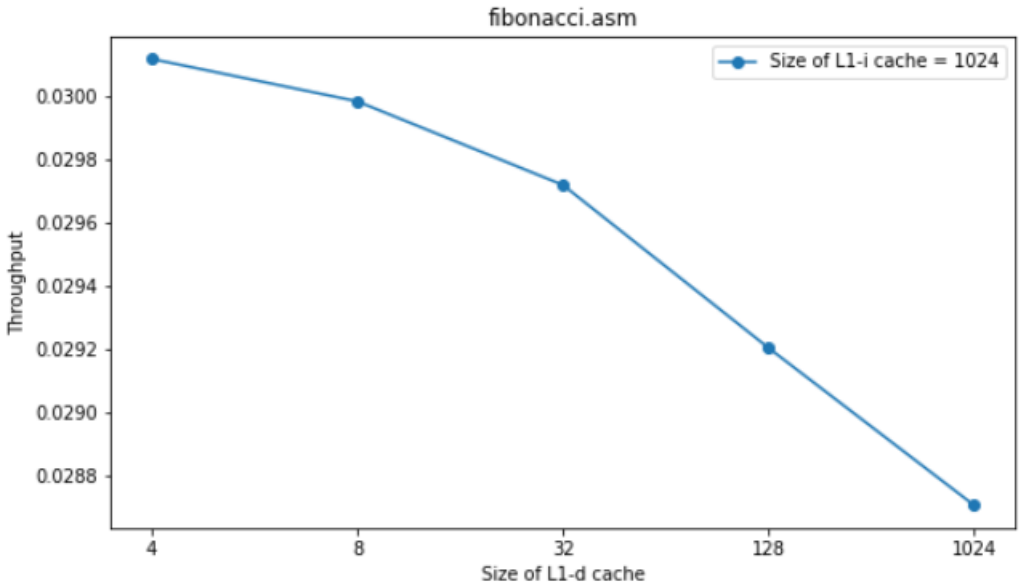




- In evenodd.asm, we observe that there are no loops in the code, hence the use of cache is practically useless.
- Hence as cache size increases, overall number of cycles increase, thus reducing thorughput.

2. Fibonacci.asm

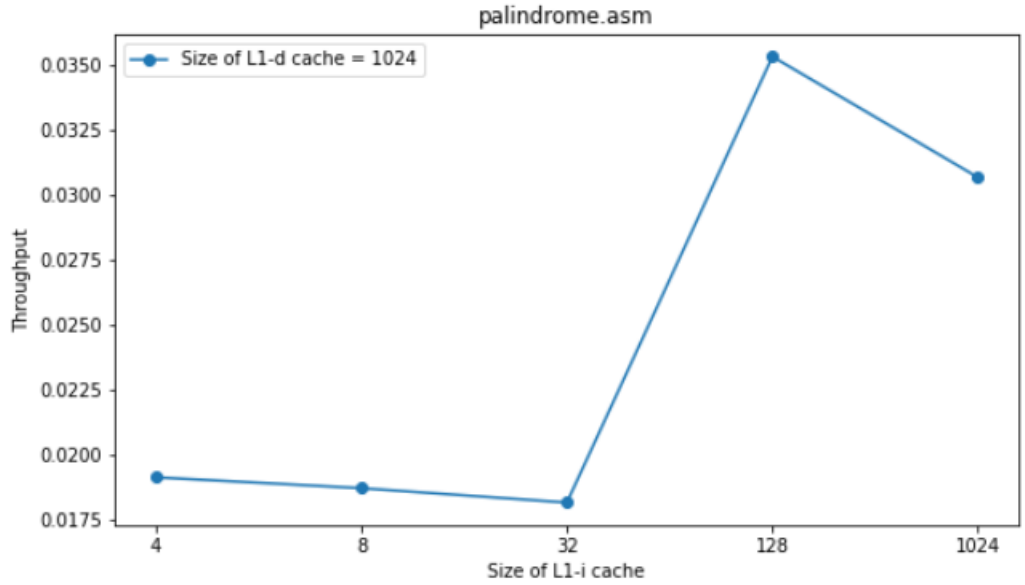


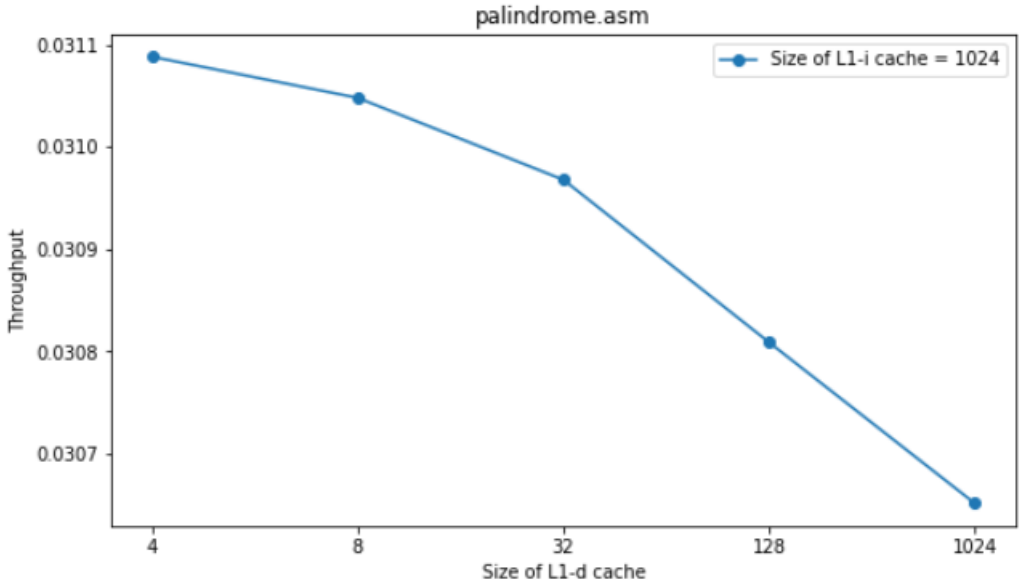


- In Fibonacci.asm, since we have a considerable number of instructions in the loop, we need a sufficiently large L1-i cache to store the instructions. This sufficiently large size is attained at 128B, and hence we observe a peak in the throughput in the first graph.

- In the case of L1-d cache, we do not perform any MemoryRead operations in the loop, hence the size of the L1-d cache would not positively impact the throughput.

3. palindrome.asm

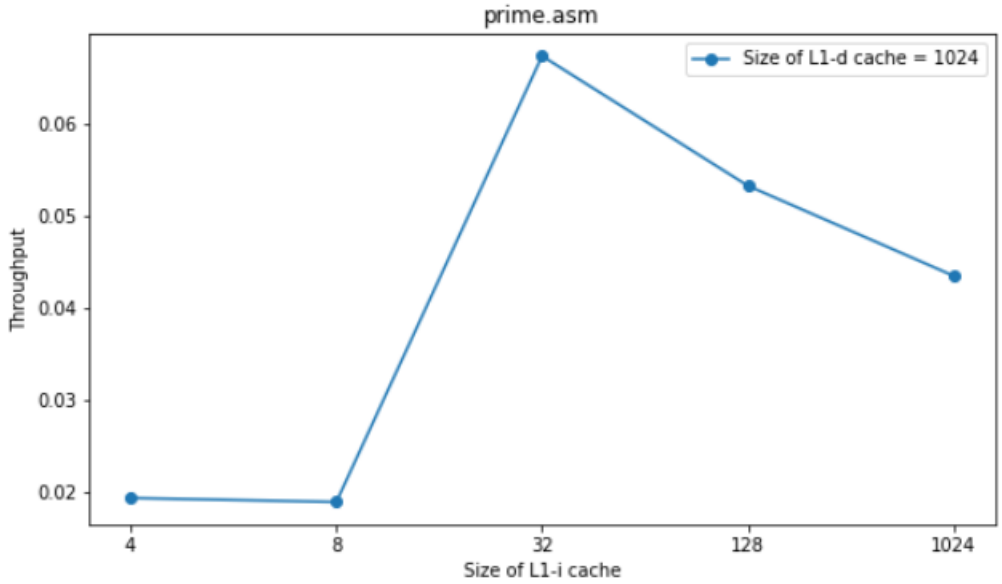


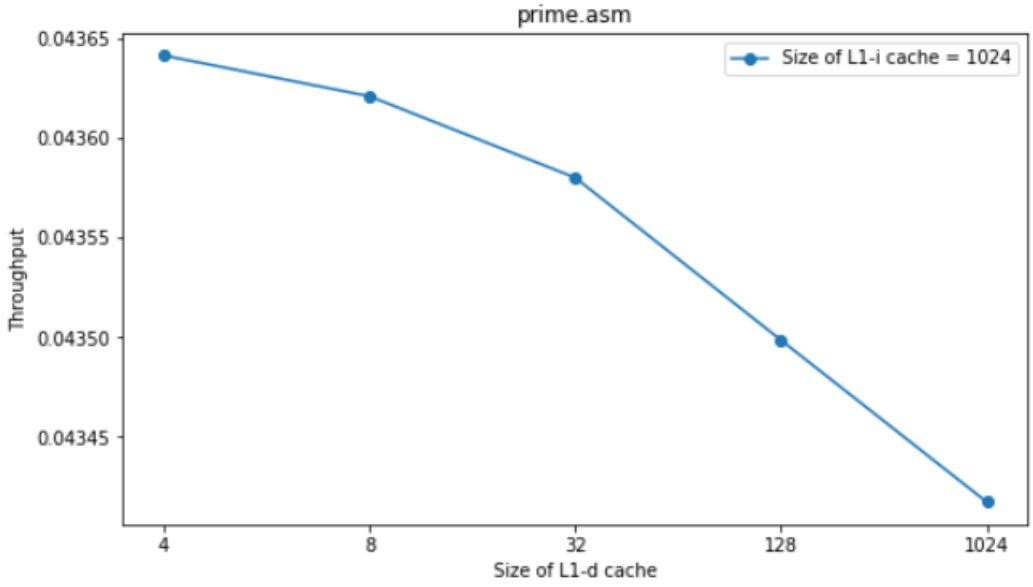


- Similarly in palindrome.asm, since we have a considerable number of instructions in the loop, we need a sufficiently large L1-i cache to store the instructions. This sufficiently large size is attained at 128B, and hence we observe a peak in the throughput in the first graph.

- In the case of L1-d cache, we do not perform any MemoryRead operations in the loop, hence the size of the L1-d cache would not positively impact the throughput.

4. prime.asm

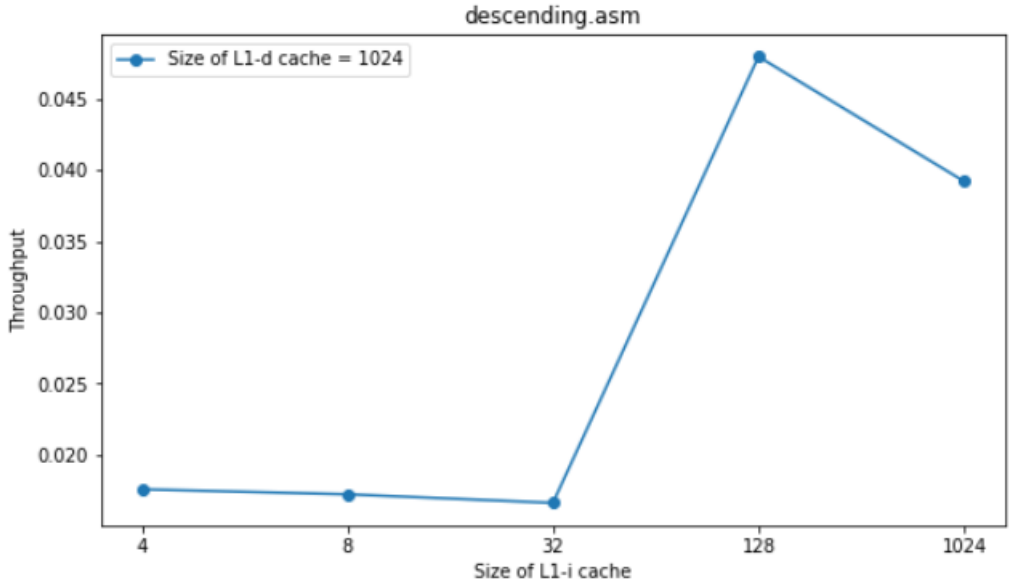


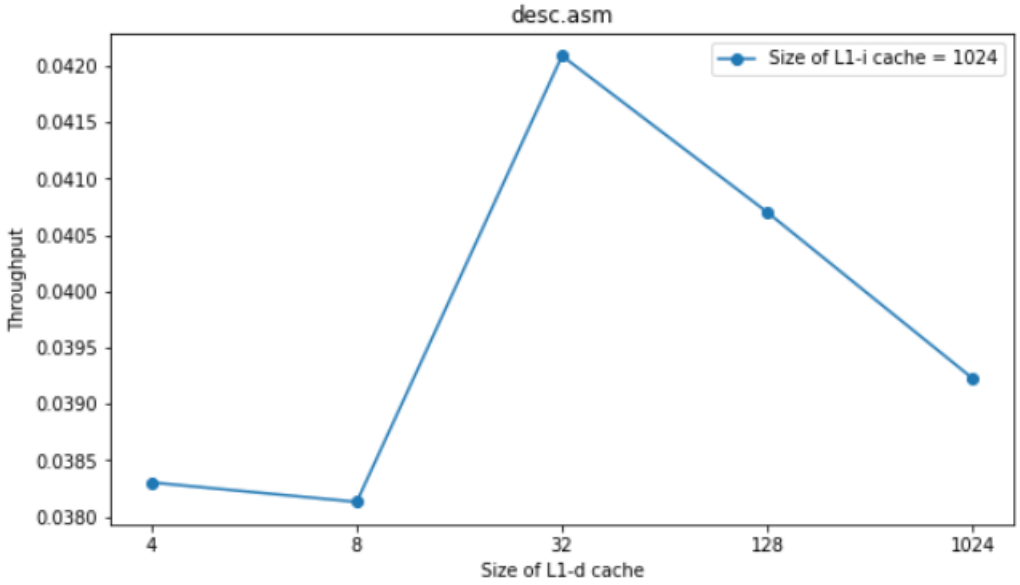


- In prime.asm, since we have a smaller number of instructions in the loop than in the previous 2 benchmarks, we need a comparatively smaller L1-i cache to store all the instructions. This size is attained at 32B, and hence we observe a peak in the throughput. On increasing the size further, we observe a decline in the throughput as there is no advantage in increasing the L1-i cache size.

- In the case of L1-d cache, we do not perform any MemoryRead operations in the loop, hence the size of the cache would not positively impact the throughput.

5. descending.asm





- In descending.asm, since we have a large number of instructions in the loop, we need a large L1-i cache to store all the instructions. This optimal size is attained at 128B, and hence we observe a peak in the throughput. On increasing the size further, we observe a decline in the throughput as there is no advantage in increasing the L1-i cache size.

- In the case of L1-d cache, we have a few MemoryRead operations (load), which accounts for the peak observed at 32B, as the L1-d cache now stores all the required data. On increasing the size further, we observe a decline in the throughput as there is no advantage in increasing the L1-d cache size.