COL 216 assignment 3
Report

The goal was to design a cache simulator having two cache levels L1 and L2. The simulator uses LRU cache eviction policy and WBWA cache write policy.

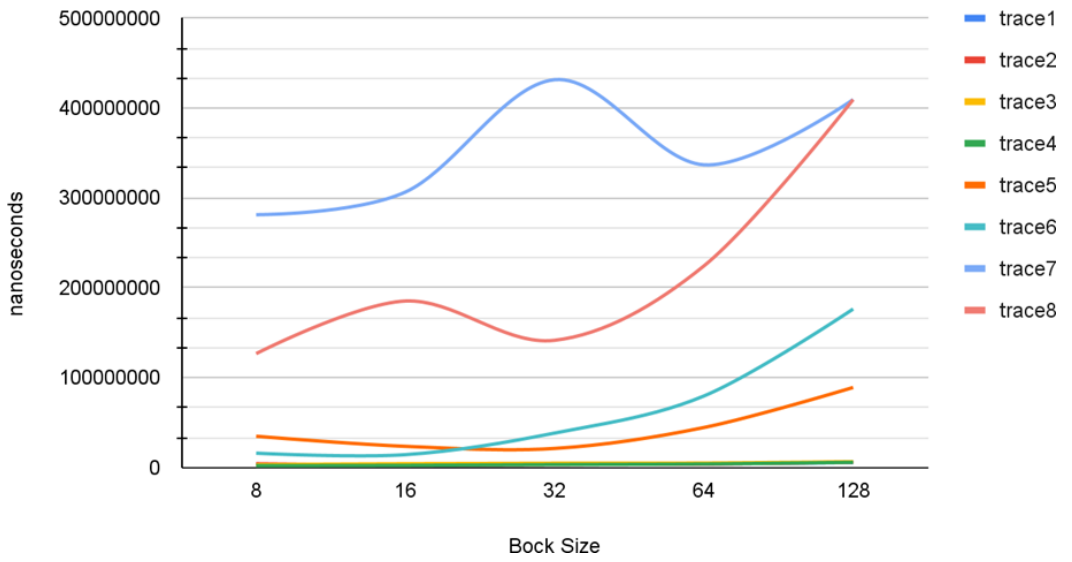
The C++ simulator takes the following parameters :

- BLOCKSIZE
- L1_SIZE
- L1_ASSOC
- L2_SIZE
- L2_ASSOC

File	Instructions
trace1	100000
trace2	100000
trace3	100000
trace4	100000
trace5	4000000
trace6	5200000
trace7	8500000
trace8	9600000

The following graphs show us the effect of varying these parameters on the total response time by running the simulator on the given trace files. We only vary one parameter at a time in the x axis while keeping all others constant . The default values are **64, 1024, 2, 65536, 8**. The y axis represents total access time. Since we have 8 trace files, each graph will contain 8 lines.

- ## Graph 1
  We vary **block size** on x-axis between 8, 16, 32, 64, 128.



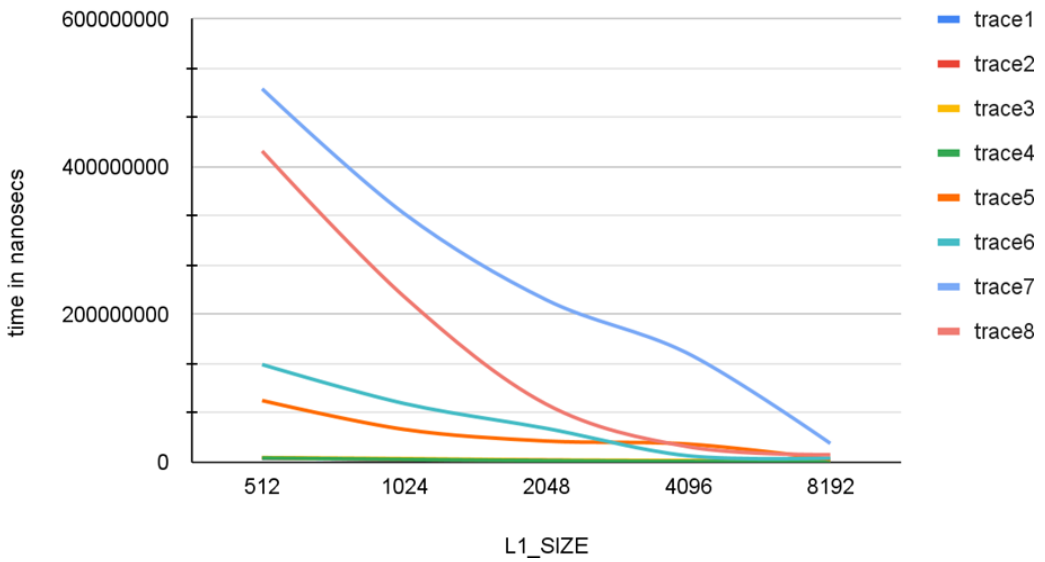
- We see that trace1, trace2, trace3 and trace4 show similar plots . Each file has 100000 instructions.
- trace5 shows a dip at block size 32 whereas trace6 shows a dip at block size 16. Comparing their plots we conclude trace5 uses better temporal spatiality then trace6 overall.
- For trace7 a block size of 32 leads to significant rise in access time implying that it makes the least utilisation of temporal locality.
- For trace8 a block size of 8 or 16 is preferable as the make the best use of temporal locality.
- Although trace8 is larger than trace7 , it takes less time at all block sizes implying lesser misses and better use of locality (next access address often in same cache in trace8 compared to trace7)
- Increasing the block size of a cache can improve performance by enhancing spatial locality and reducing compulsory misses. However, beyond a certain point, it can lead to cache pollution, increased conflict misses, and longer transfer times, resulting in an increase in access time. There is a trade-off between block size and access time, and the optimal block size depends on cache design and workload characteristics.

Graph 2

We vary the second parameter **L1 size** between 512, 1024, 2048, 4096 and 8192.

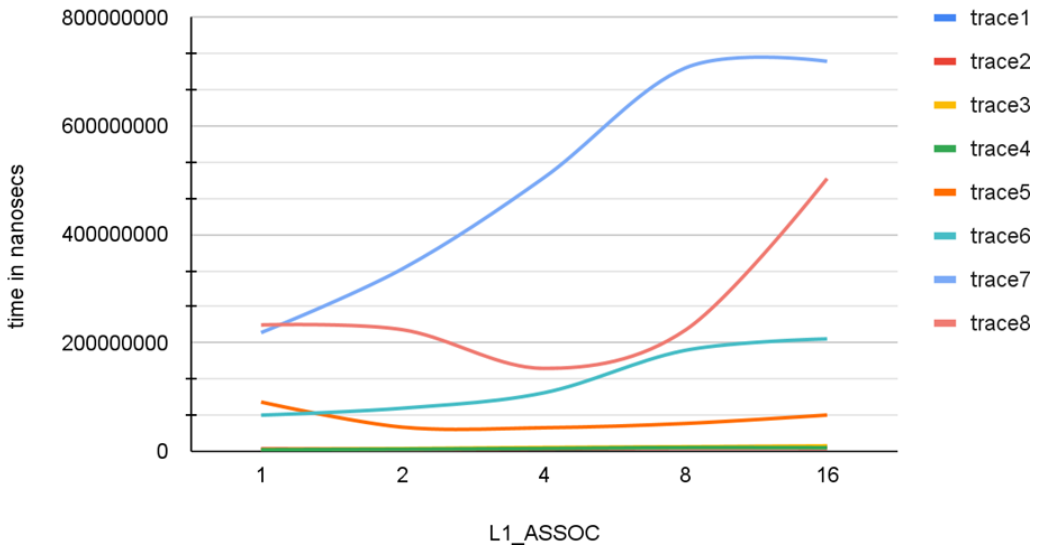
- We see that trace1, trace2, trace3 and trace4 show similar plots . Each file has 100000 instructions.
- Decrement in access time is faster in case of trace6 as compared to trace5 implying that increasing L1 cache size leads to lesser conflicts for trace6.
- Increase in L1 cache size leads to significant decrement in access time due to less conflicts for trace7 and trace8 .

- Decrement is more in trace 8 with the slope stabilisation at 4096 compared to trace7 implying better use of locality in trace8.
- Although trace8 is larger than trace7 , it takes less time at all L1 cache sizes implying lesser misses and better use of locality (next access address often in same cache in trace8 compared to trace7)
- Increasing L1 cache size leads to decrease in access time for all traces due to the principle of locality (specifically temporal locality) and reduced cache misses.



- ## Graph 3

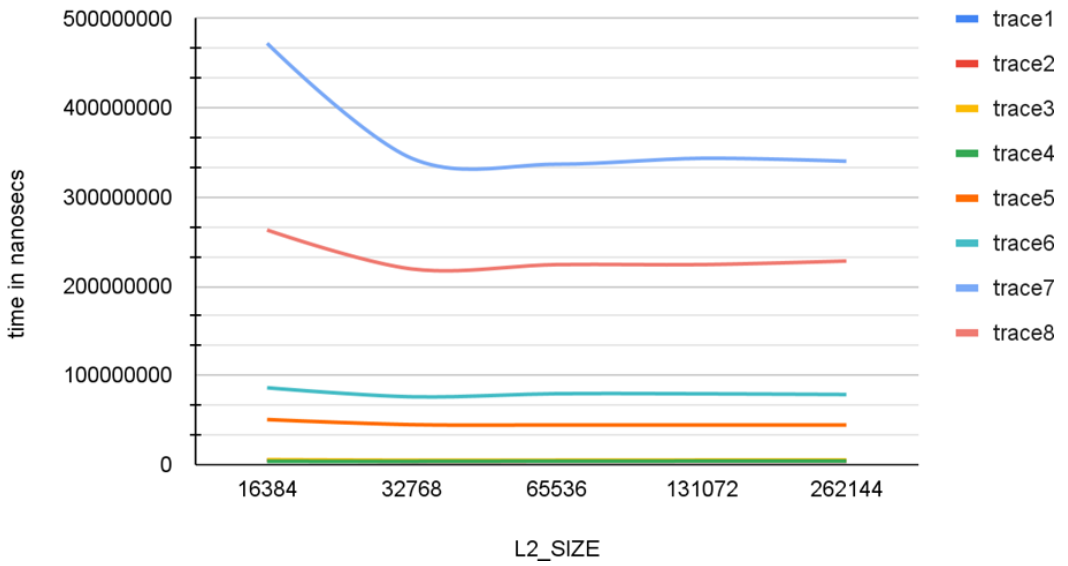
We vary the third parameter **L1 associativity** between 1, 2, 4, 8 and 16.



- We see that trace1, trace2, trace3 and trace4 show similar plots . Each file has 100000 instructions.
- For trace5 and trace8 ,L1 associativity of 4 shows a dip implying it leads to lesser conflicts as it makes best utilisation.
- For trace6 and trace7 increasing L1 associativity leads to no improvement in access time showing addresses accessed next don't have the same tag bits often leading to no use of it.
- Increasing L1 associativity beyond 4 leads to increase in access time for all traces as when the associativity of an L1 cache is increased, the number of cache sets decreases, and more cache blocks can potentially map to the same set. This reduction in the number of cache sets leads to an increase in the number of cache conflicts.

Graph 4

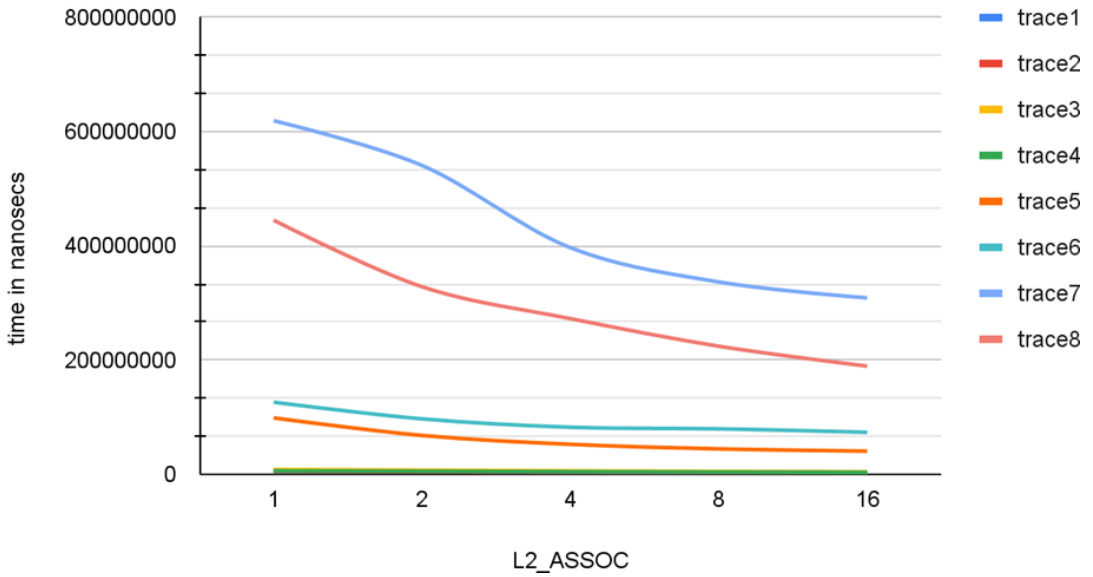
We vary the fourth parameter **L2 size** between 16384, 32768, 65536, 131072 and 262144.



- We see that trace1, trace2, trace3 and trace4 show similar plots i.e. no change in access time . Each file has 100000 instructions.
- For trace5 and trace6, no significant change observed by changing L1 cache size
- For trace7 and trace8 increasing L2 cache size beyond 32768 leads to stagnation of access time.
- Going from L2 cache size of 16384 to 32768 shows decrement for trace8 and trace7 as in this case larger L2 cache can store more data, reducing the likelihood of cache conflicts and improving cache hit rates.
- Increasing the size of the L2 cache typically does not have a significant effect on access time due to the hierarchical structure of the memory system.

- **Graph 5**

  We vary the fifth parameter **L2 associativity** between 1, 2, 4, 8 and16.



  - We see that trace1, trace2, trace3 and trace4 show similar plots. Each file has 100000 instructions.
  - For trace5 and trace6, slight decrement is observed with increasing L2 associativity.
  - For trace7 and trace8, decent decrement is observed with increasing L2 associativity.
  - Increasing the associativity of the L2 cache can lead to a decrease in access time due to improved cache hit rates and reduced cache conflicts.

Token Distribution:

Total tokens to be distributed = 20

Name	Entry Number	Tokens
Kanishka Gajbhiye	2021CS50131	15
Himadri Rajora	2021CS10117	5