CS230 PROJECT

MEMORY/CACHE HIERARCHY OPTIMIZATIONS FOR SAT SOLVERS

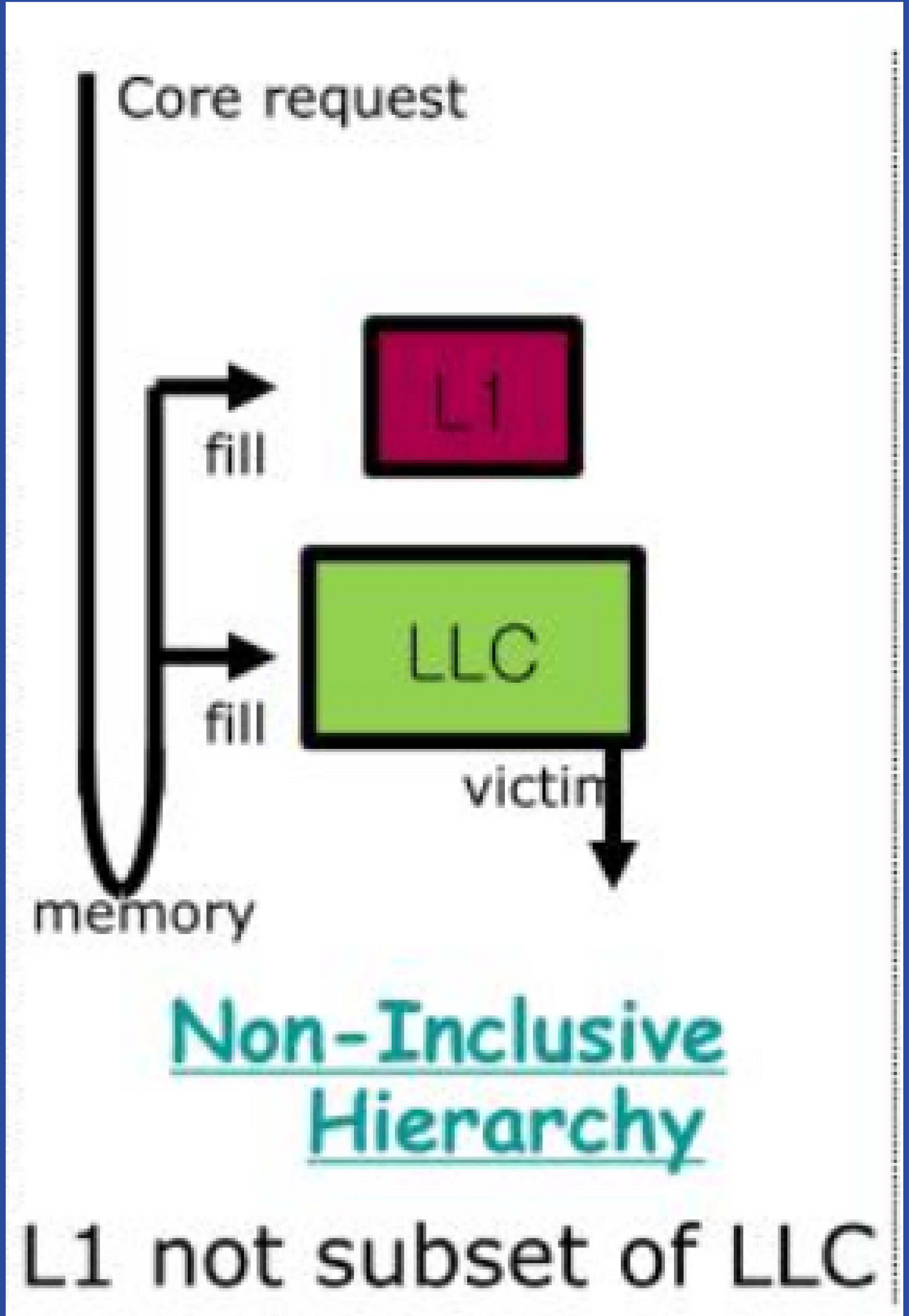
GASADA SREELAXMI 210050052
NARKEDAMILLI HARIKA 210050168
MONINGI SRIJA 210050098
YARABOLU VENNELA 210050168

MOTIVATION

- TASK: TO EVALUATE AND COMPARE DIFFERENT CACHE HIERARCHIES WITH VARYING SIZES OF L1, L2, AND LLC, AS WELL AS DIFFERENT CACHE REPLACEMENT POLICIES.
- WE ARE ALSO EXPECTED TO COMPARE THEIR PERFORMANCE WITH A BASELINE CACHE HIERARCHY AND SUGGEST WAYS TO IMPROVE CACHE PERFORMANCE BASED ON OUR ANALYSIS.
- THE TRACES FOR TWO SAT SOLVERS, MULTIPLE CADICAL AND KISSAT, HAVE BEEN PROVIDED TO GET CONCLUSIONS..

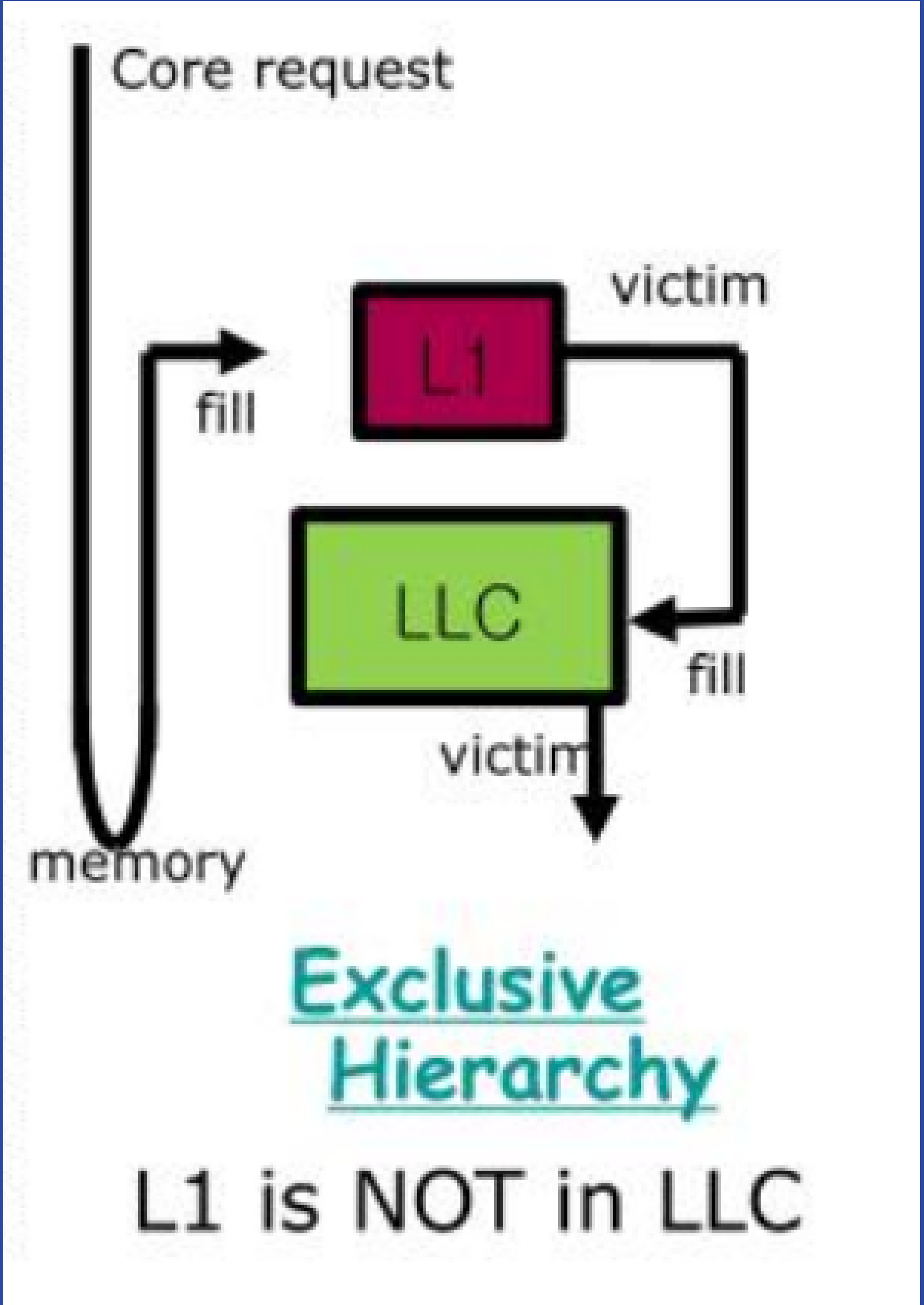
NON-INCLUSIVE

Under a non-inclusive cache policy, blocks should be filled into both the LLC cache (public cache) and the private L1/L2 cache. There is no interdependence between the caches when it comes to evicting blocks out of them



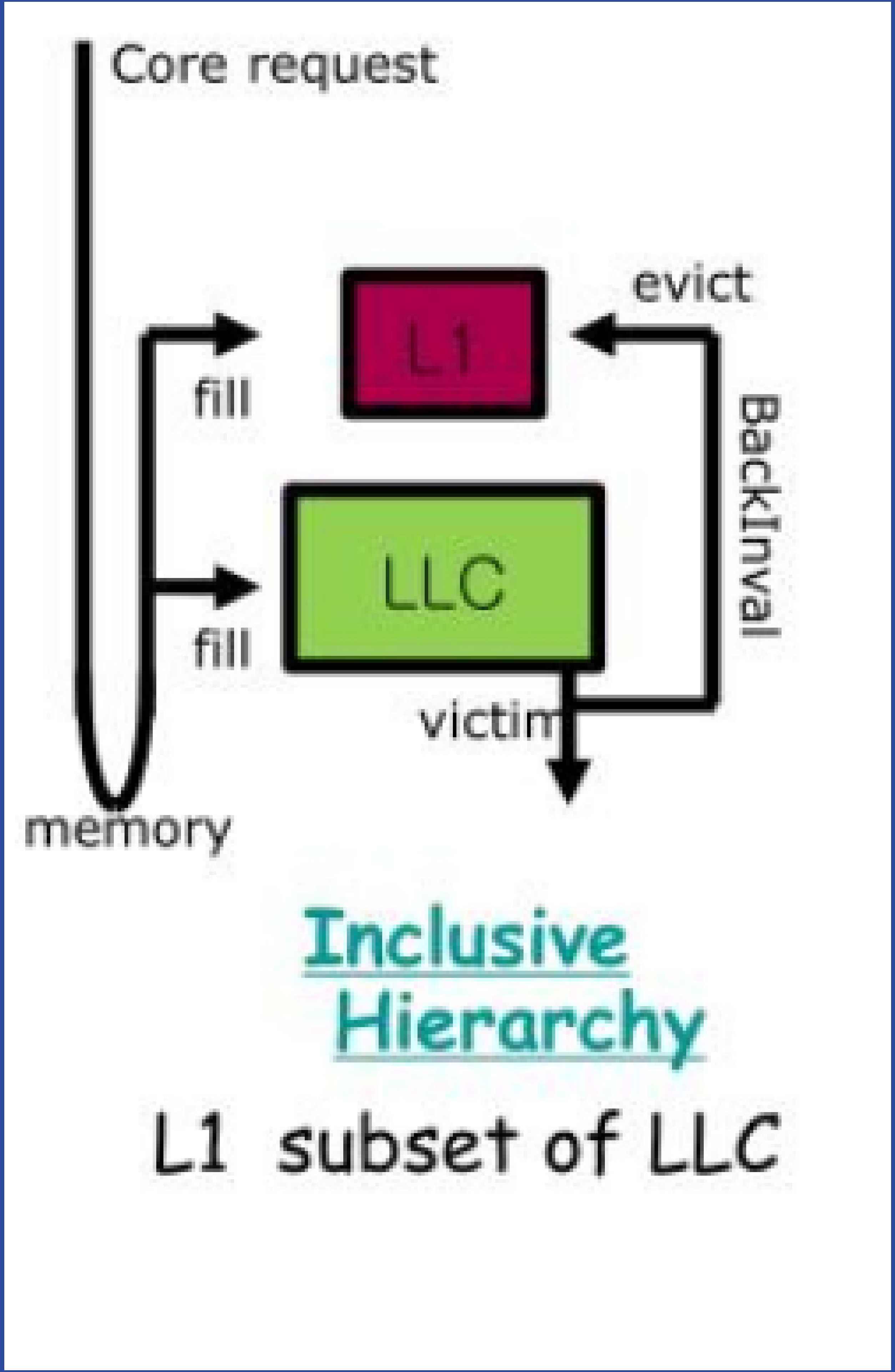
EXCLUSIVE

An exclusive cache policy ensures that when a block is filled into the cache, it is only stored in either the L1/L2 cache or the LLC cache, but not both. If a block is evicted from the L1/L2 cache, it will be moved to the LLC cache. In other words, a block cannot exist simultaneously in both the L1/L2 cache and the LLC cache at the same time under the exclusive cache policy.

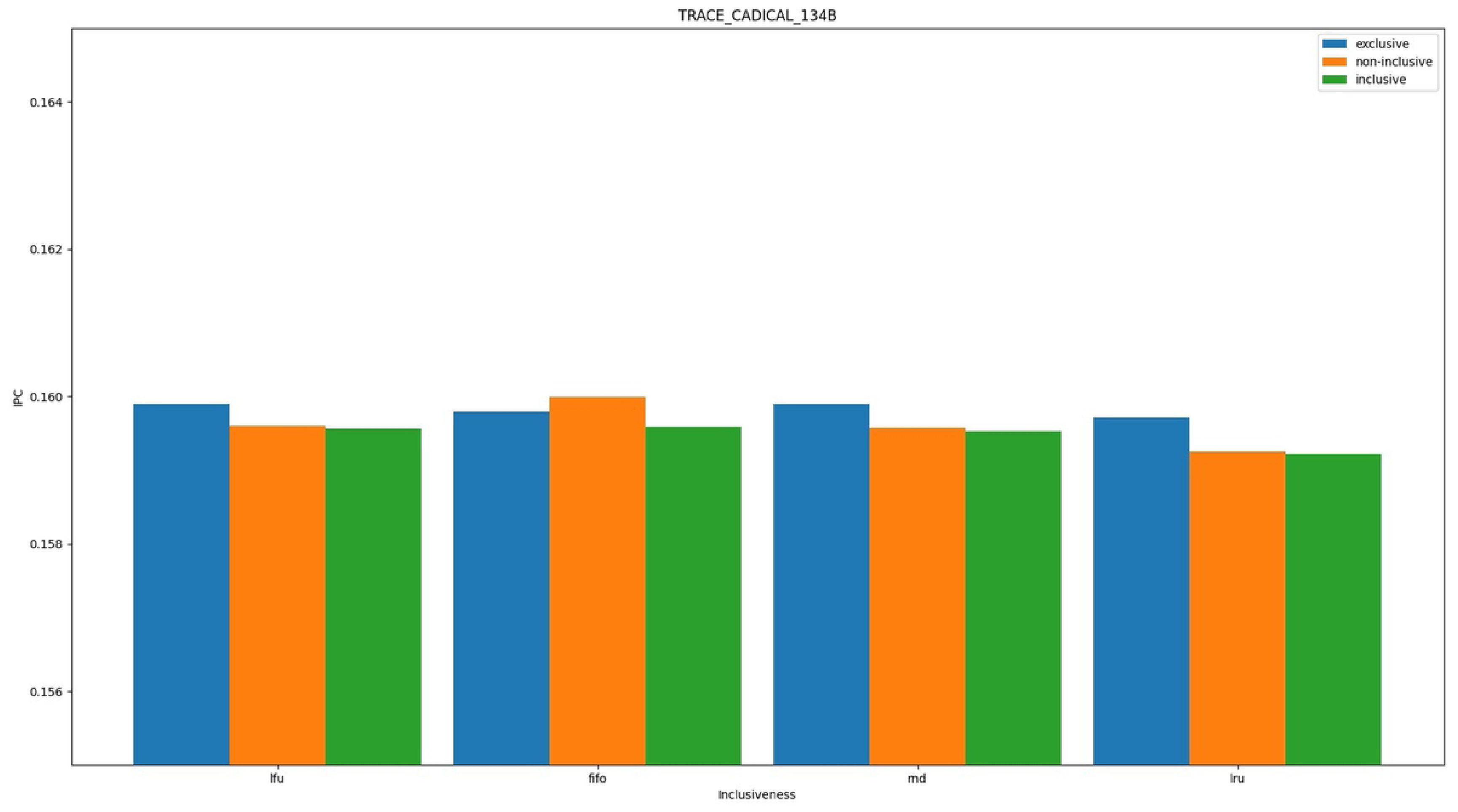


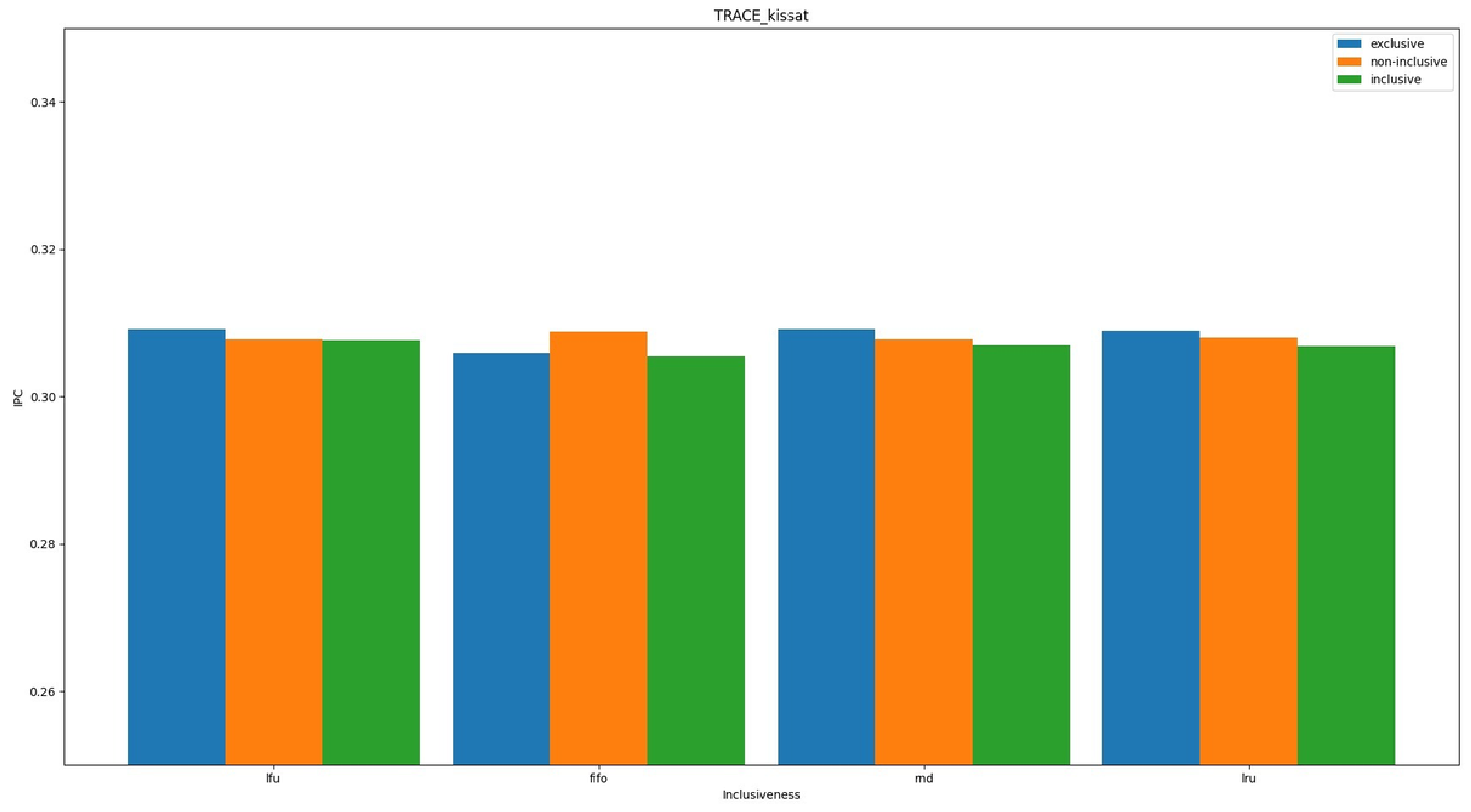
INCLUSIVE

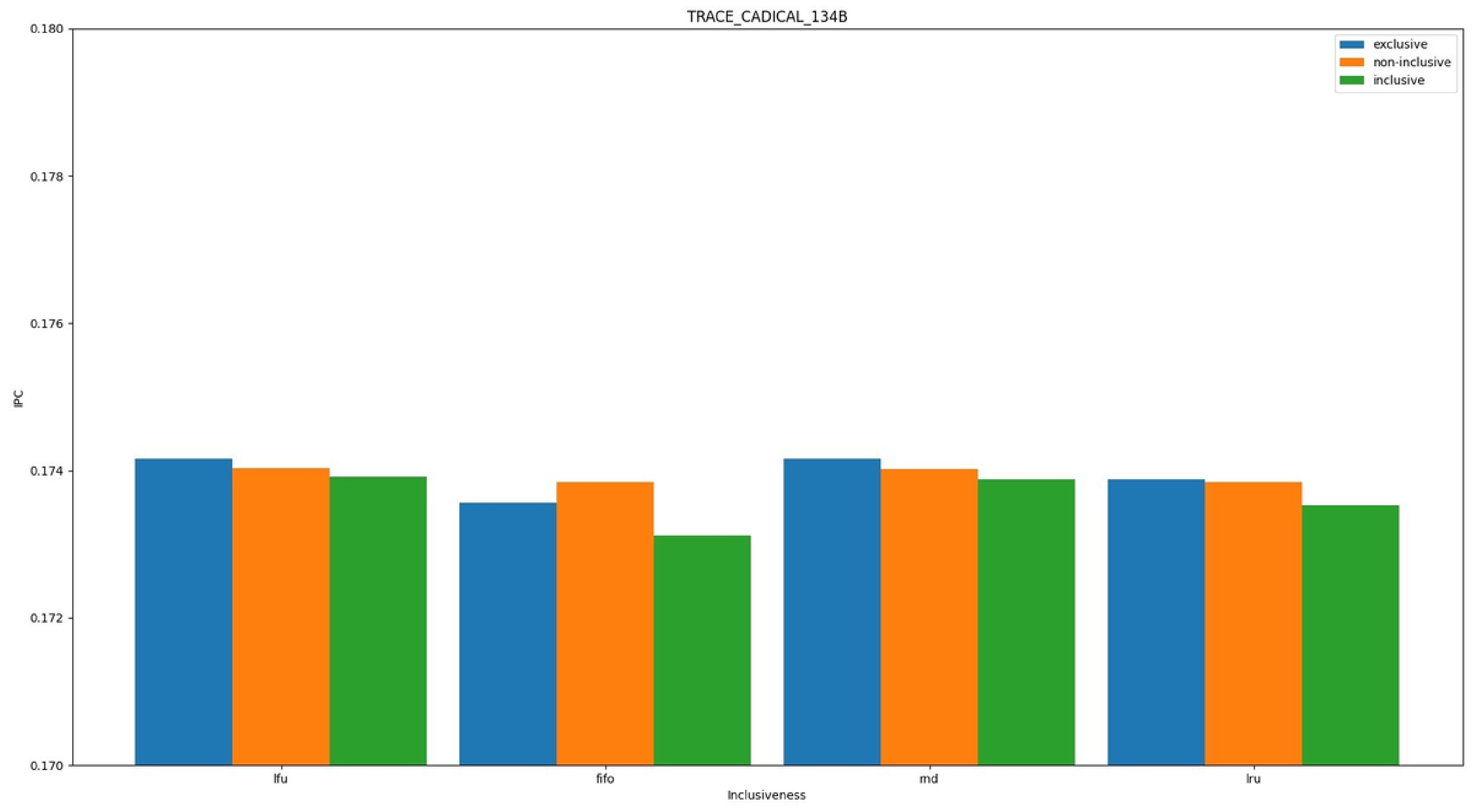
- Data is present in the L1/L2 cache if and only if it is present in the LLC cache,
- So When a block is evicted from the LLC cache, it should be back-invalidated on all the higher-level caches to maintain the inclusive property of the LLC cache.
- LLC is superset of L1/L2 cache.



CACHE HIERARCHY comparision IPC





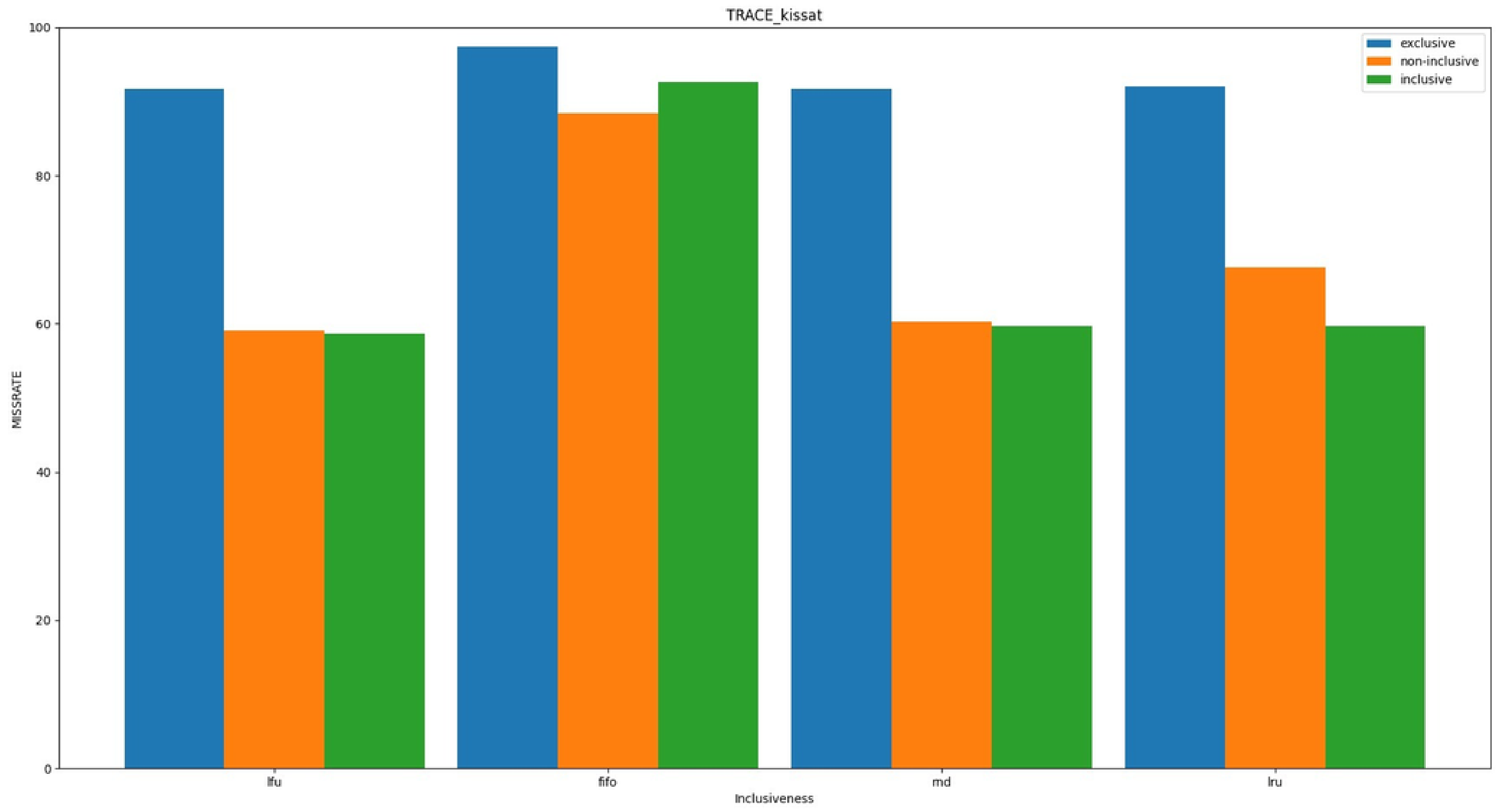


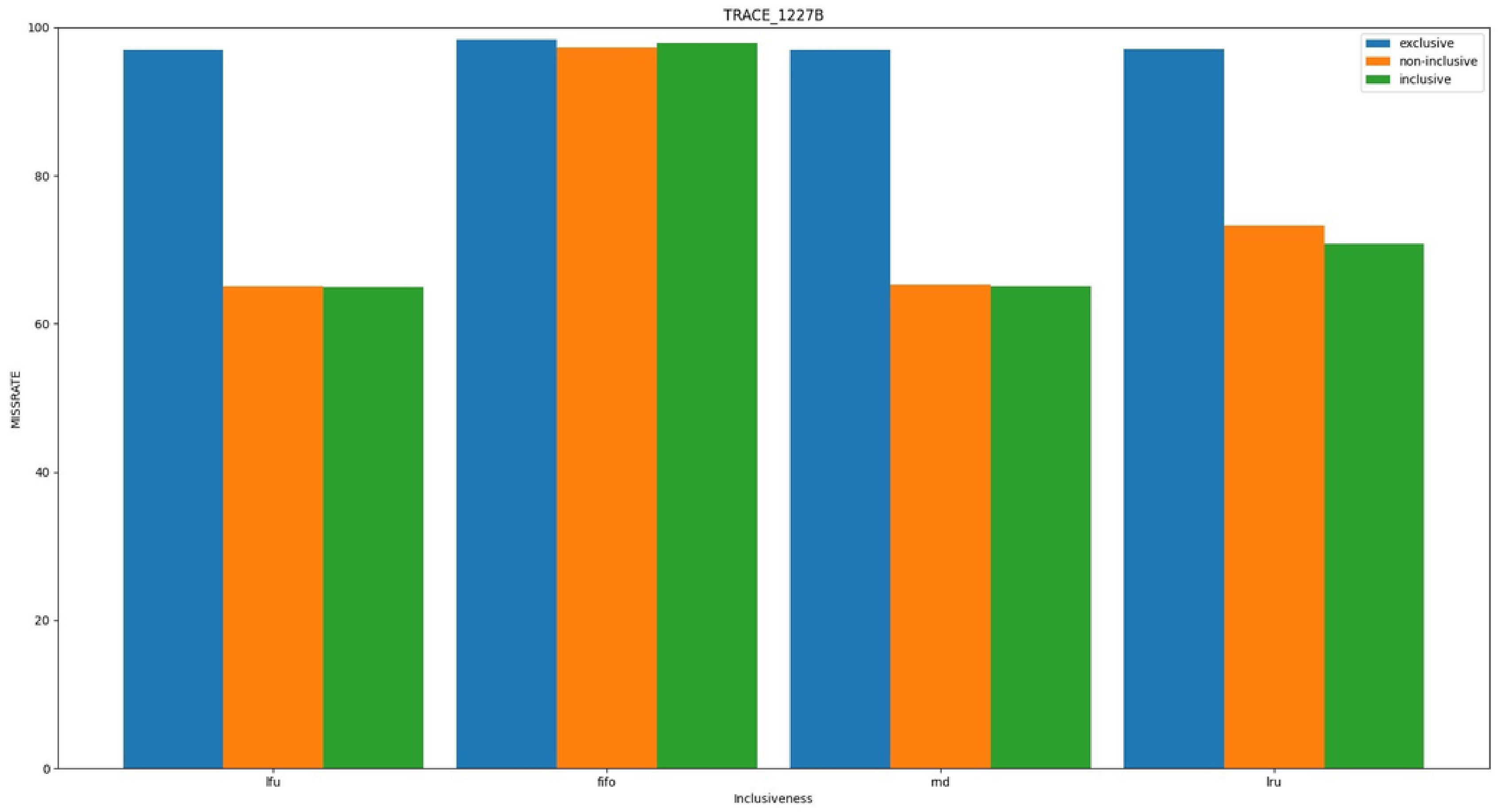
IPC:
EXC>NON_INC>INC

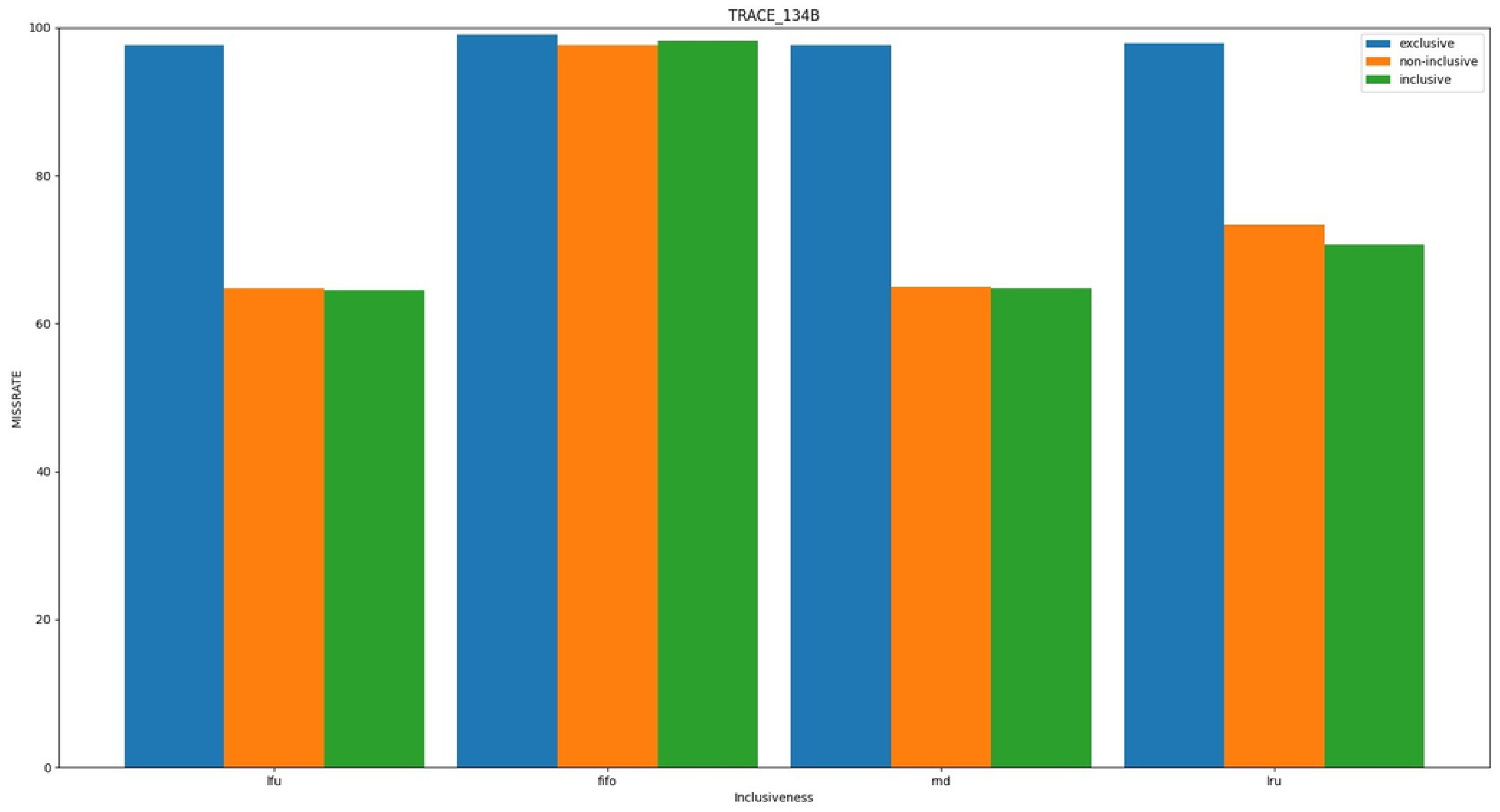
Performance:
EXC>NON_INC>INC

🟦 exclusive
🟧 non-inclusive
🟩 inclusive

CACHE HIERARCHY comparision_MISSRATE







MISS_RATE:
EXC>NON_INC>INC

Performance:
INC>NON_INC>EXC

🟦	exclusive
🟧	non-inclusive
🟩	inclusive

CACHE HIERARCHY OBSERVATIONS

ACC TO IPC:

EXCLUSIVE:

- Evicted cache line moved to another cache level.
- Other caches do not require the same data, no need to update them

NON_INCLUSIVE:

- Non-inclusive cache design contains a subset of data stored in lower-level caches.
- Cache coherence operations are reduced when a cache line is evicted from a lower-level cache and brought into a non-inclusive cache, improving IPC performance.

INCLUSIVE:

- Inclusive cache design can lower IPC performance due to maintaining copies of all lower-level cache data.
- More cache coherence operations may be needed to maintain data consistency, potentially reducing IPC performance

ORDERS

ACC TO IPC:

EXC>NON_INC>INC

ACC TO MISS RATE:

INC>NON_INC>EXC

CACHE HIERARCHY OBSERVATIONS

ACC TO MISSRATE:

The miss rate in an exclusive cache design can be higher than in non-inclusive and inclusive cache designs

For Exclusive Cache, when a cache line is evicted from one cache and brought into another cache, it is more likely that the cache line is not present in the destination cache, resulting in a cache miss.

For Non-Inclusive Cache design can have a lower miss rate since each cache contains a subset of data that is also stored in lower-level caches, reducing the need for cache coherence operations.

An inclusive cache design can also have a lower miss rate since it contains all the data stored in lower-level caches, reducing the need for cache coherence operations when a cache line is requested

PERFORMANCE ORDERS

ACC TO IPC:

EXC>NON_INC>INC

ACC TO MISS RATE:

INC>NON_INC>EXC

CACHE HIERARCHY ANALOGY

INCLUSIVE

LESS DATA
MORE MISS RATE

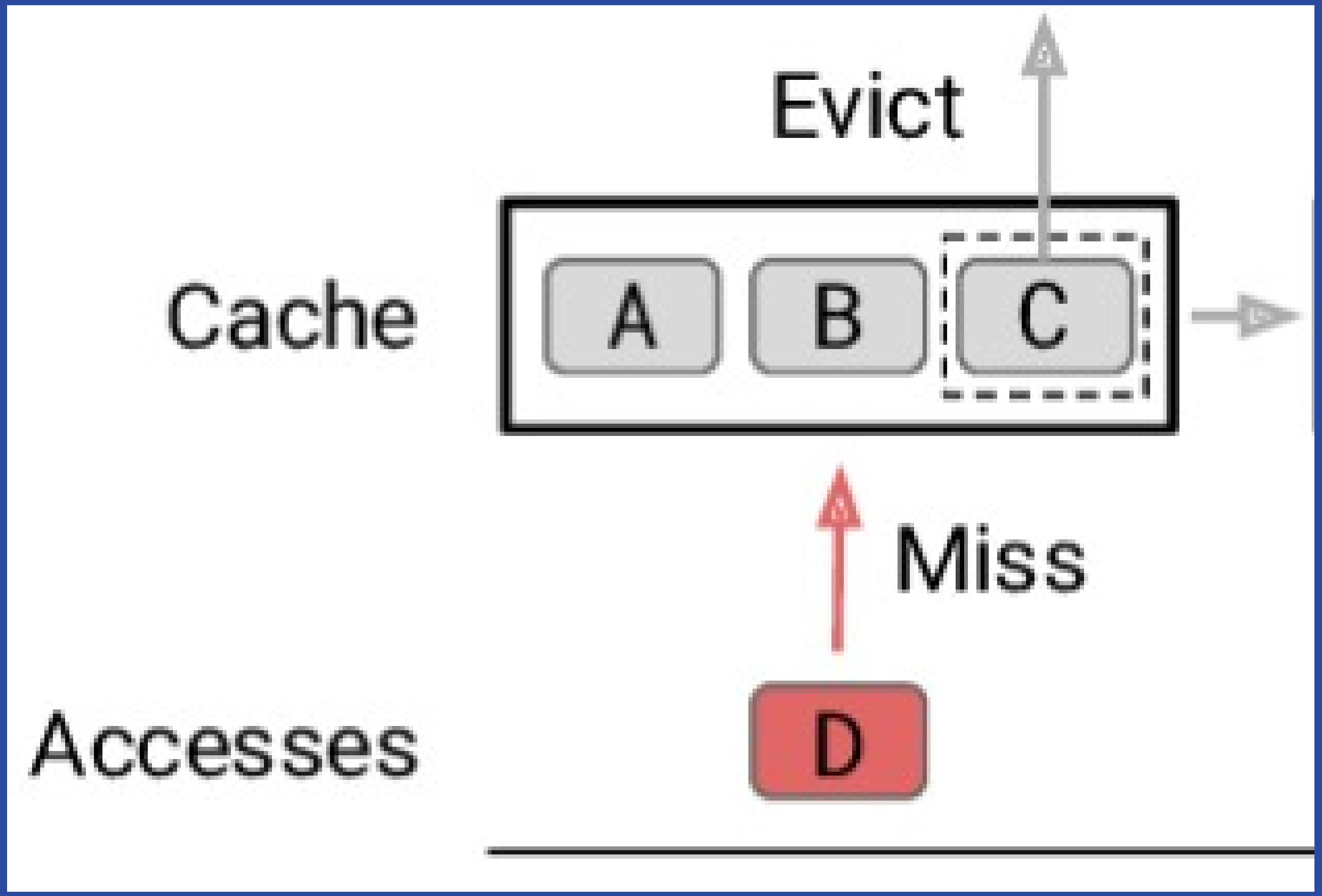
NON-INCLUSIVE

RELATIVELY MORE DATA
RELATIVELY LESS MISS RATE

EXCLUSIVE

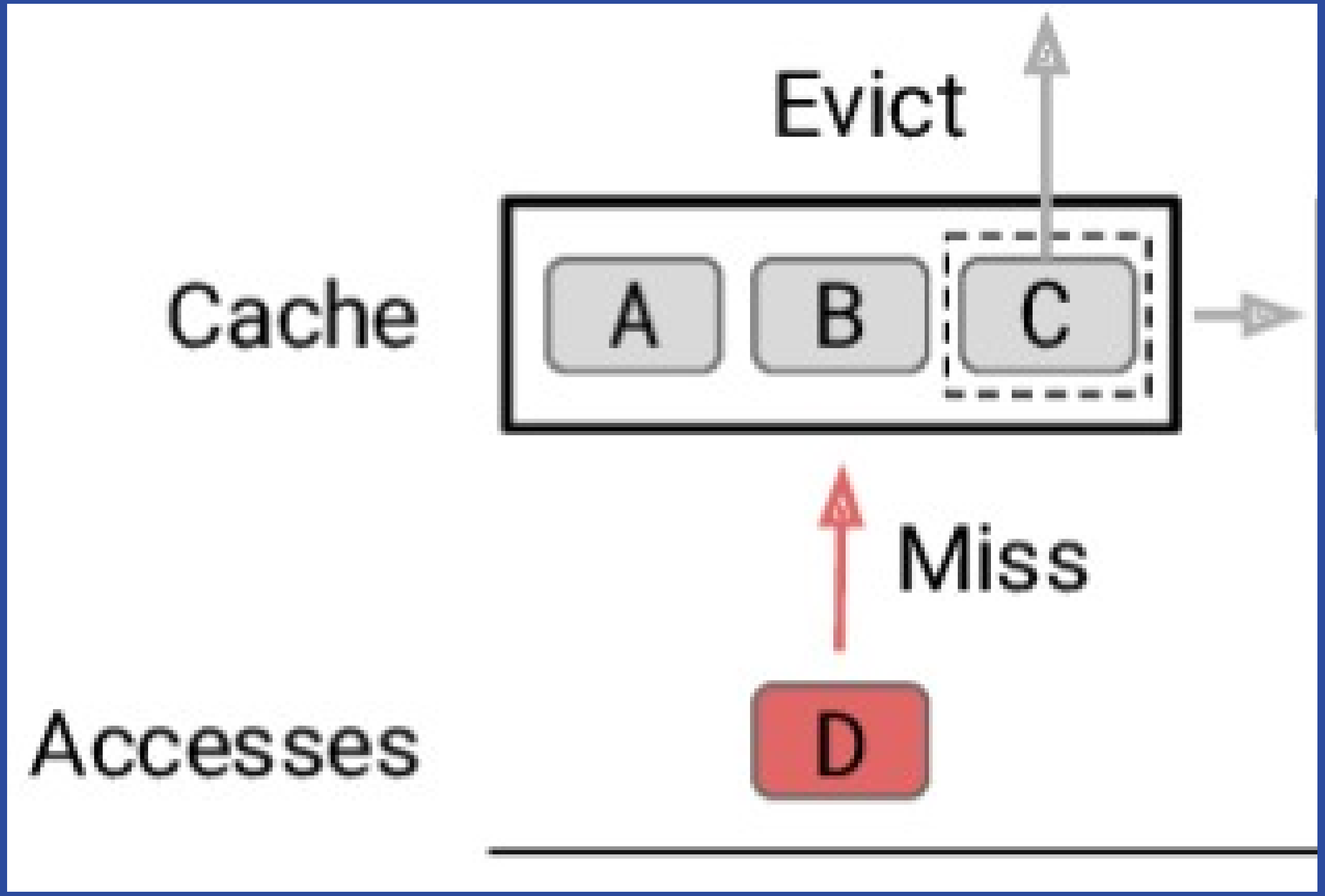
MORE DATA
LESS MISS RATE

Replacement policy



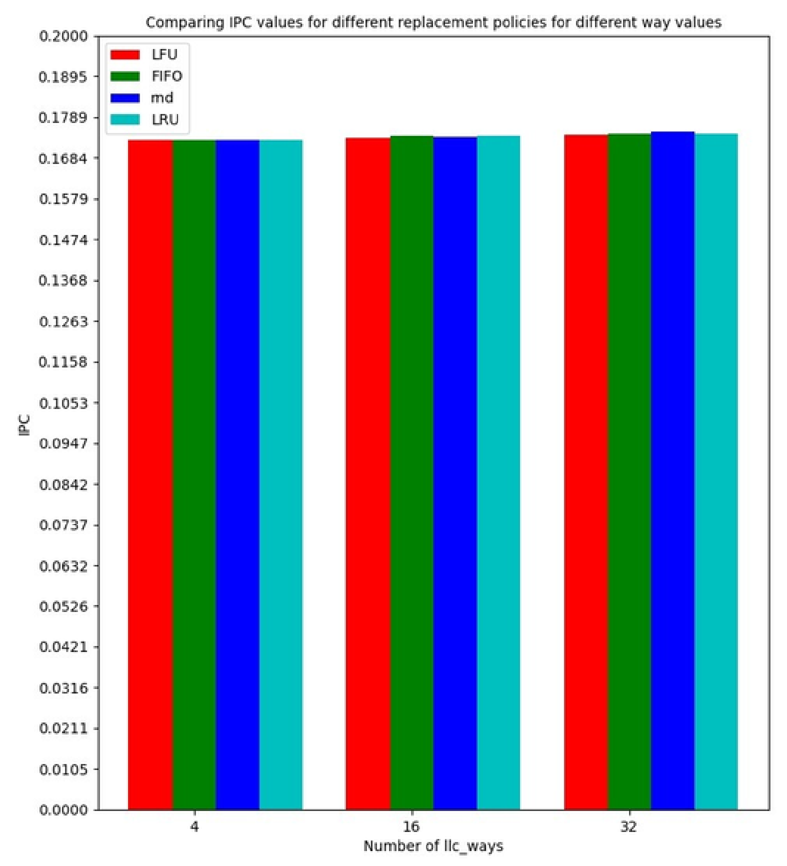
If we want to add new block into completely filled cache then we have to evict one of the old blocks and replace it with new block. So we need to prioritize the blocks and select one of them for replacement.Priorities are given to blocks according to the replacement policies like lru , lfu, fifo.

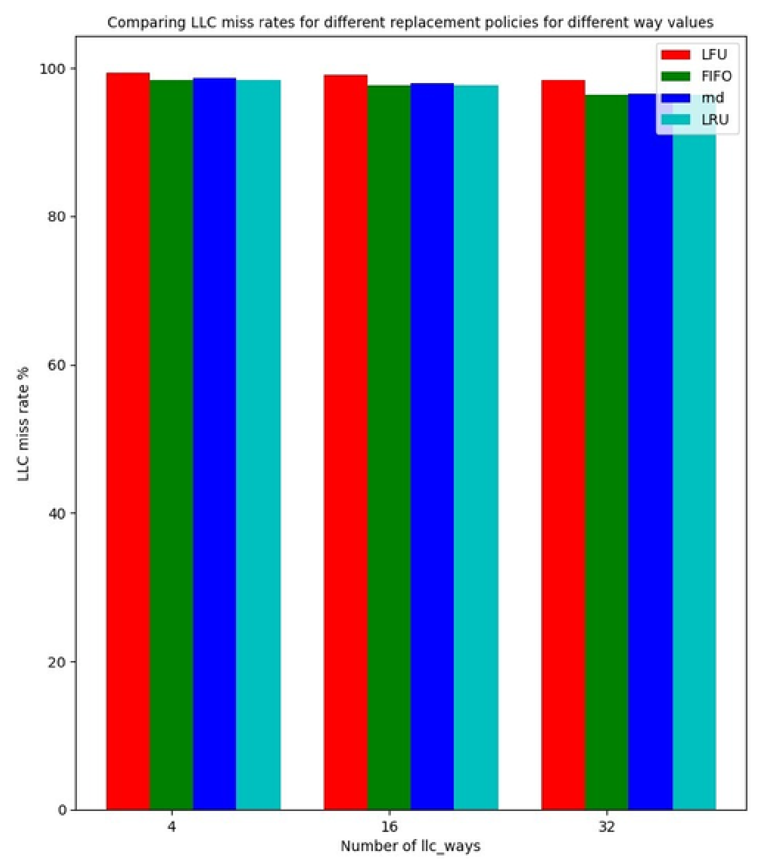
Replacement policies considered

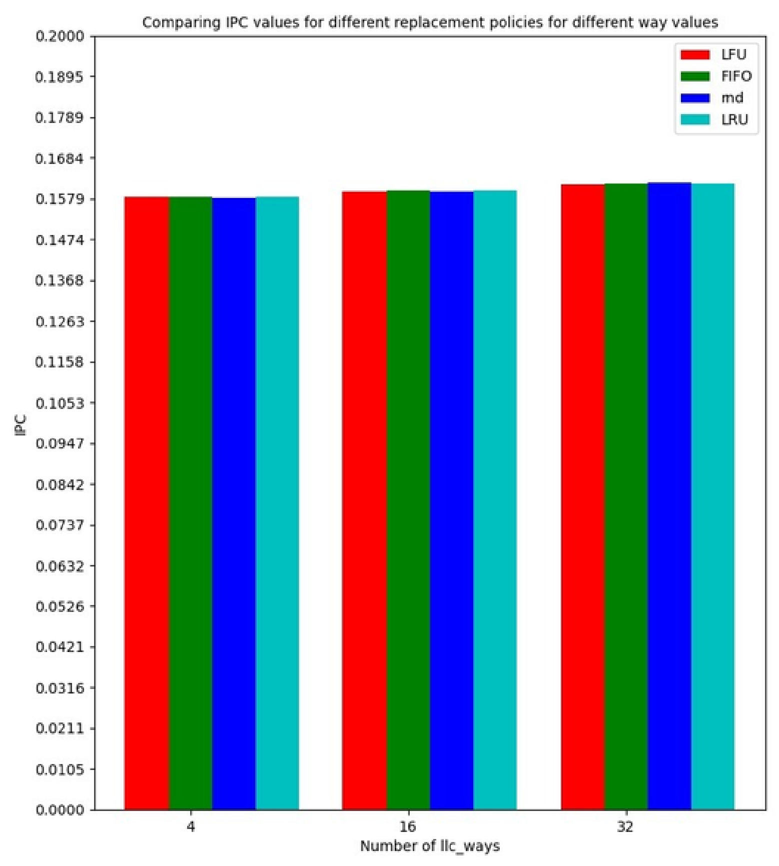


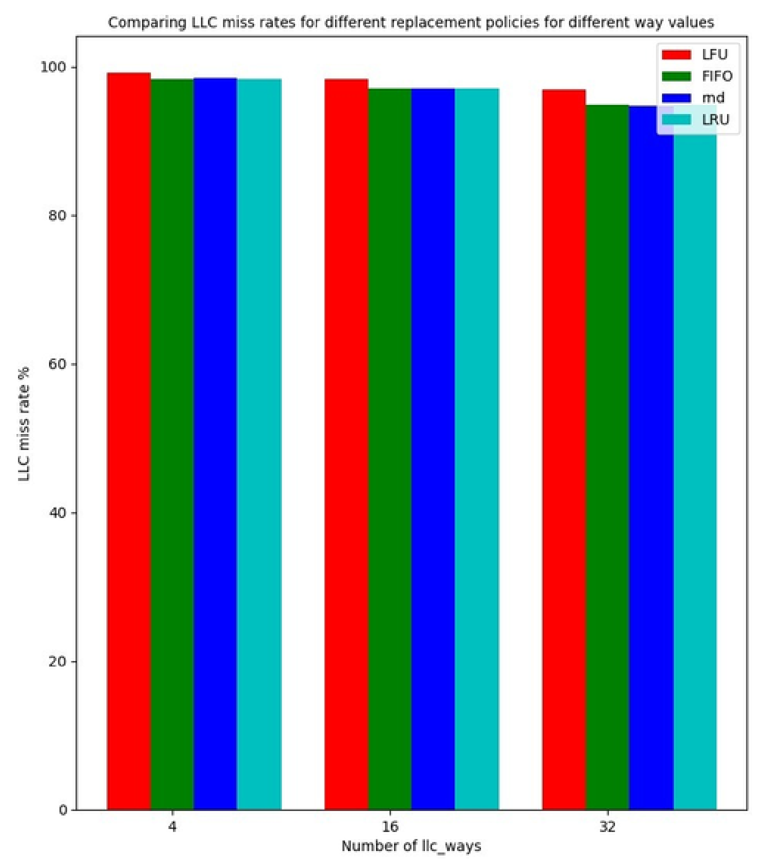
LRU	the last recently used block is evicted
LFU	least frequently used block is evicted
FIFO	oldest block is evicted
RND	block is randomly selected

Replacement comparision-EXCLUSIVE

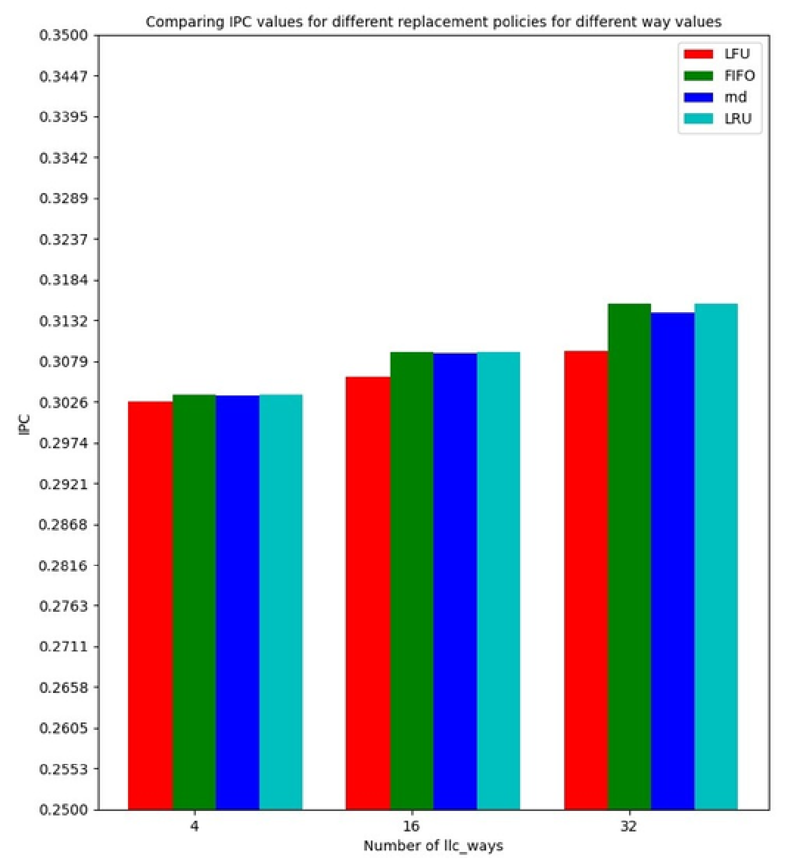


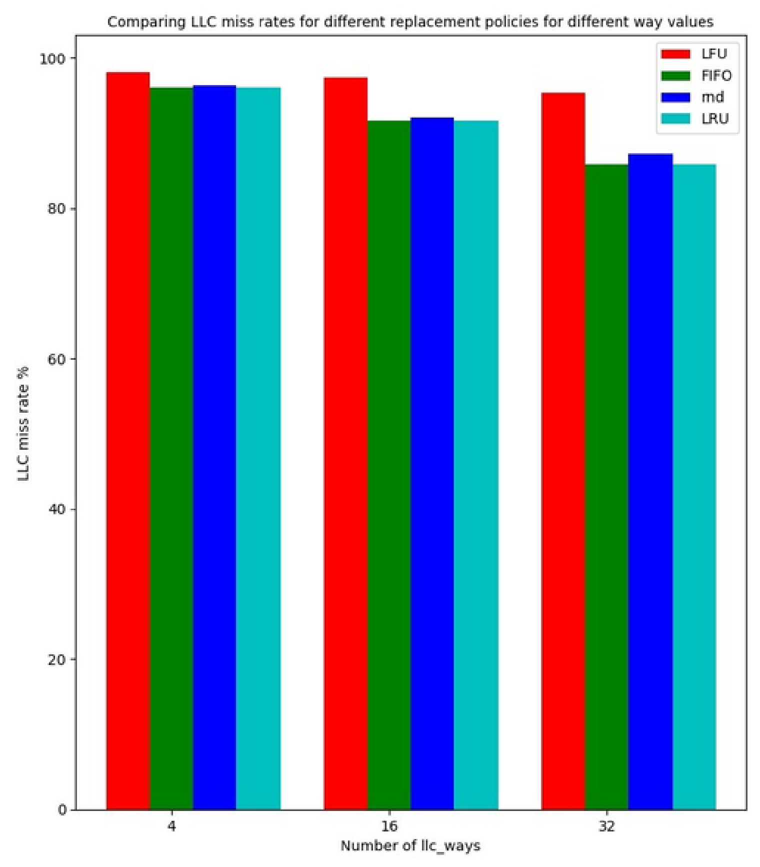






Plot for trace kissat-inc-high-30K-1802B.champsimtrace.xz

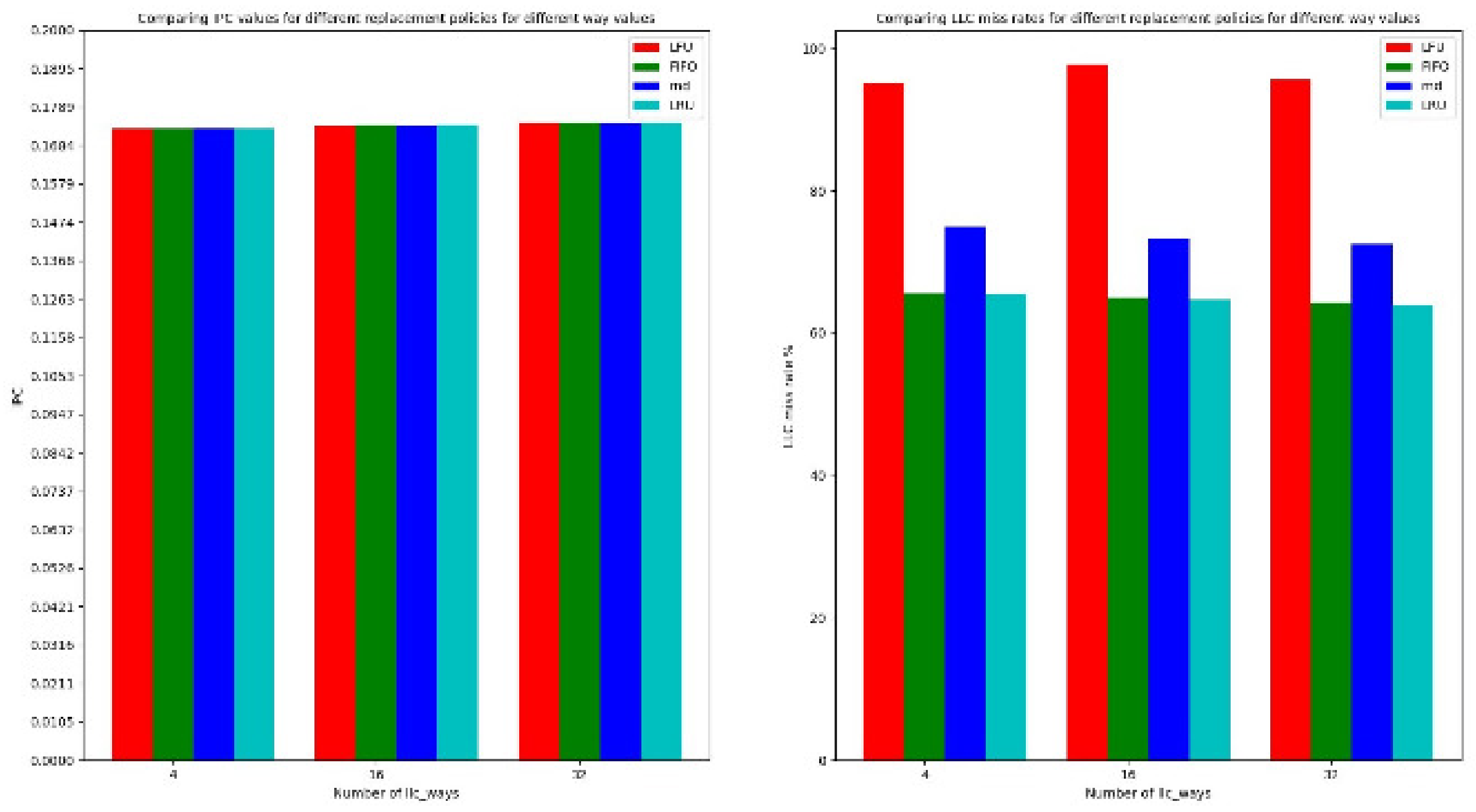




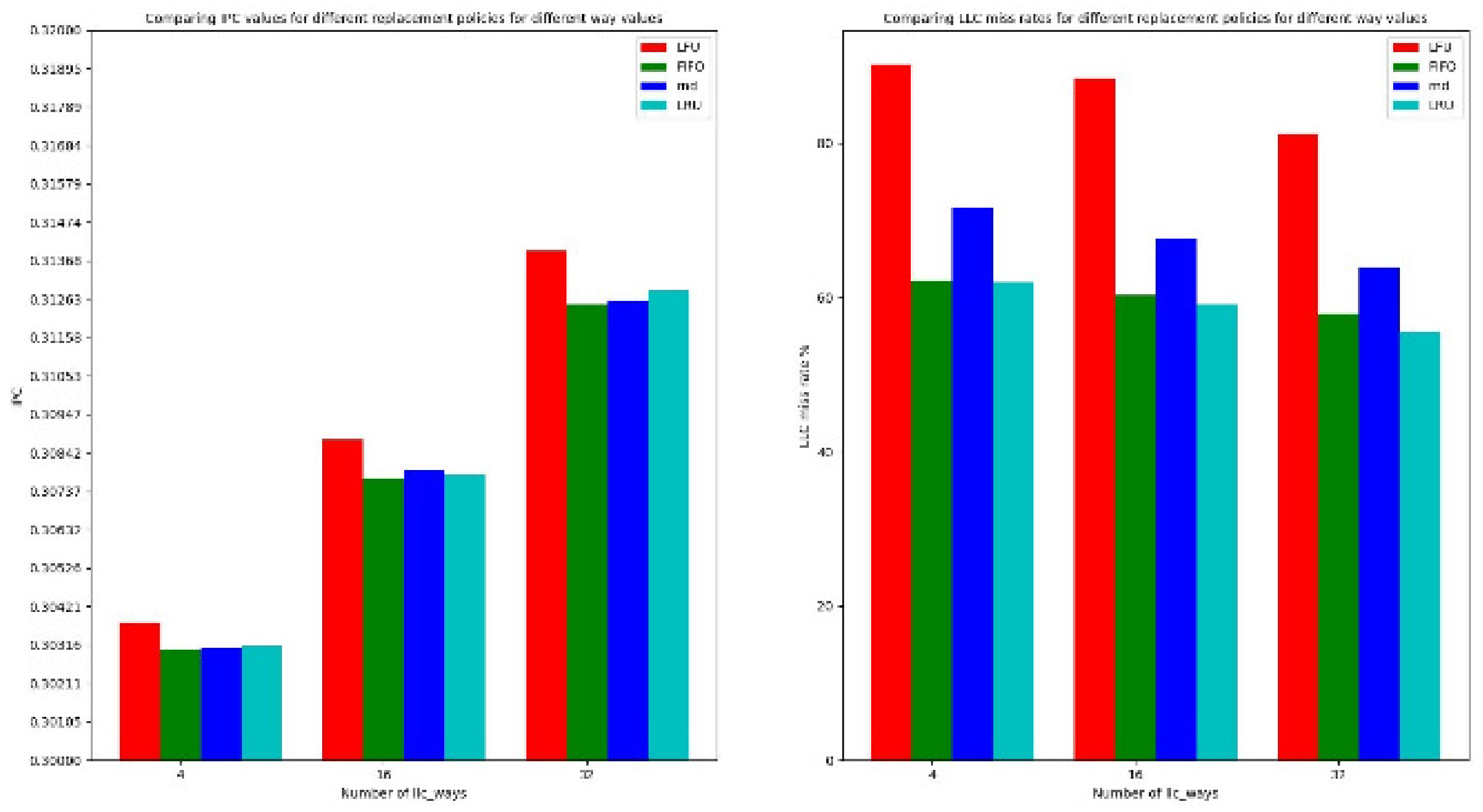
Performance(according MISS_RATE): LRU~FIFO>RND>LFU

Replacement comparision-NON-INCLUSIVE

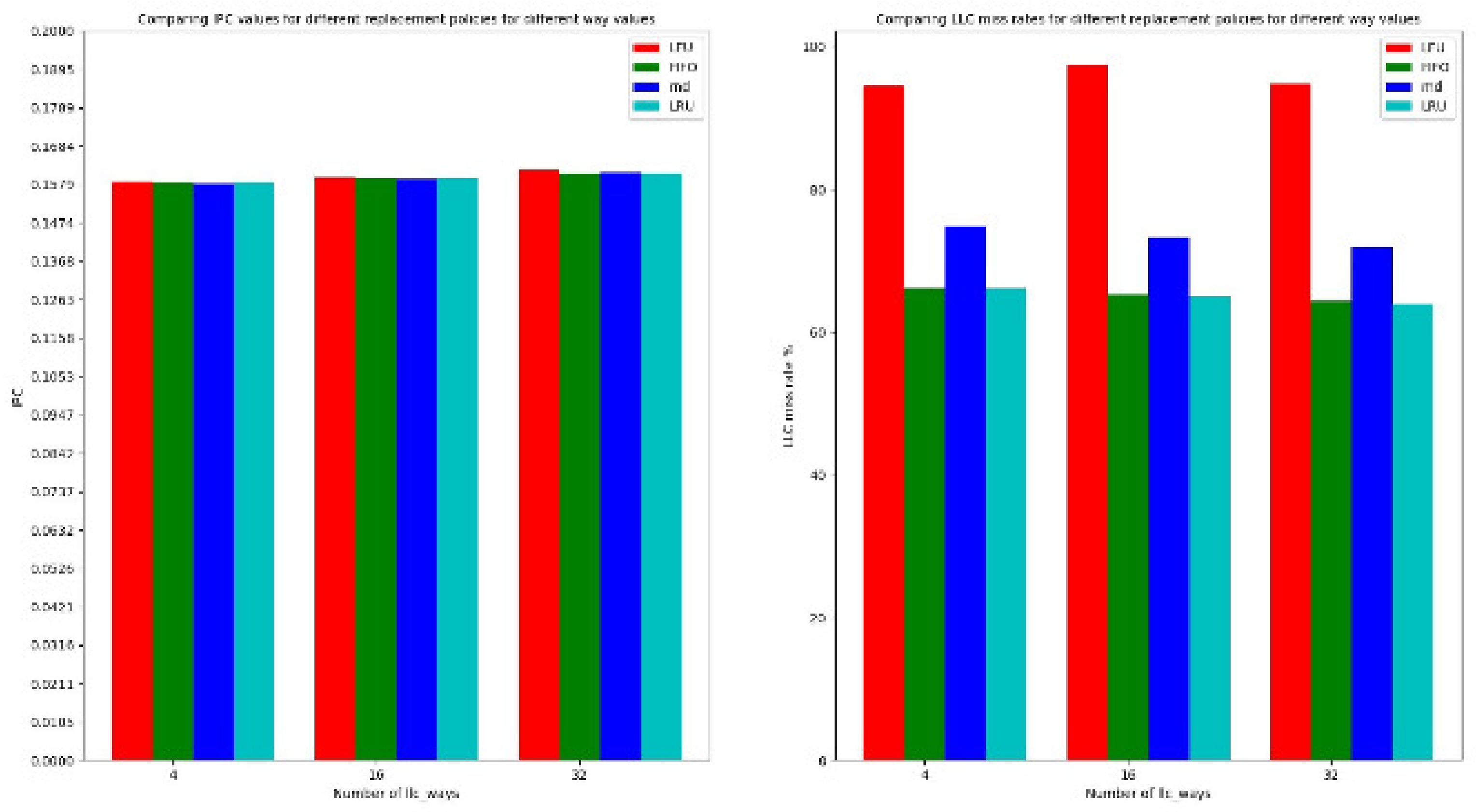
Plot for trace cadical-high-60K-134B.champsimtrace.xz



Plot for trace kissat-inc-high-30K-1802B.champsimtrace.xz



Plot for trace cadical-high-60K-1227B.champsimtrace.xz

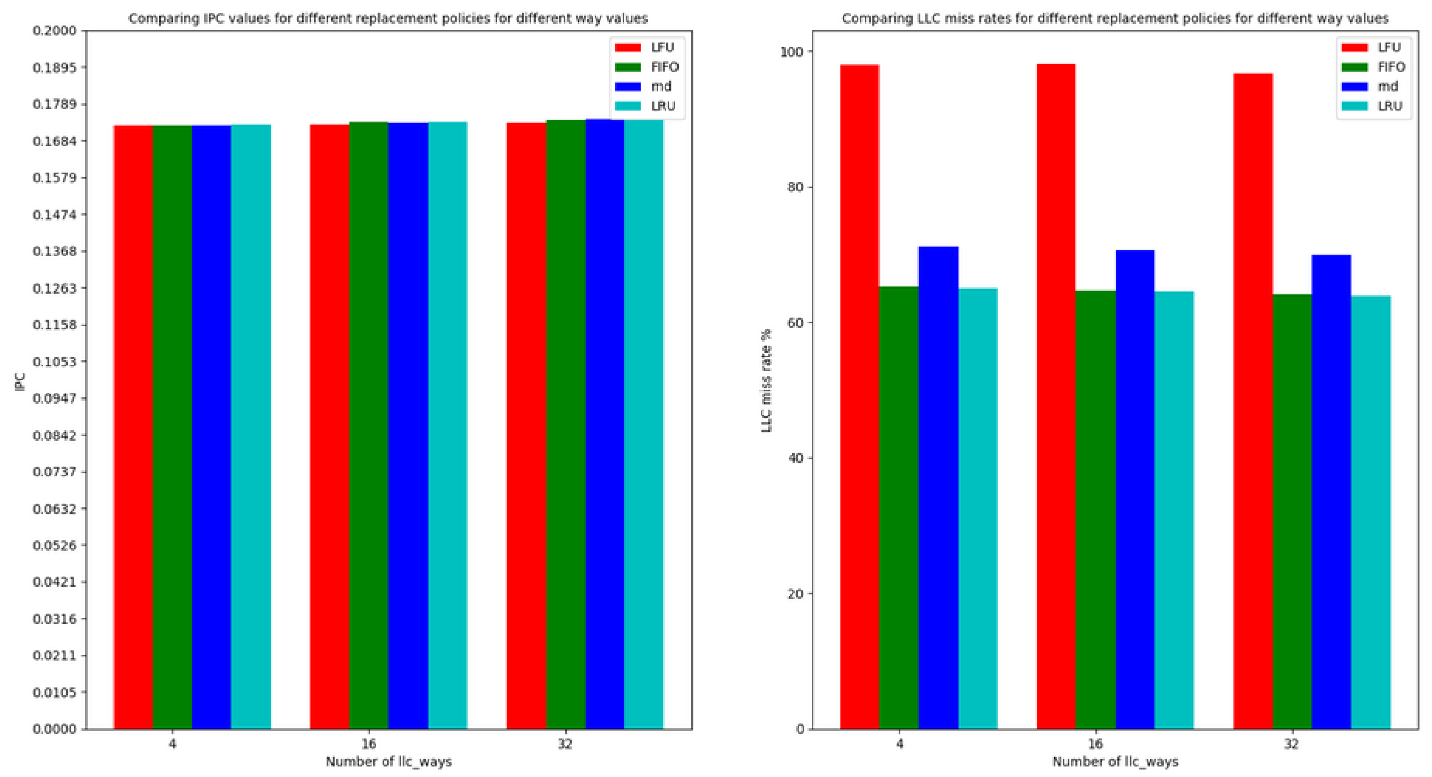


Performance(according MISS_RATE):
LRU~FIFO>RND>LFU

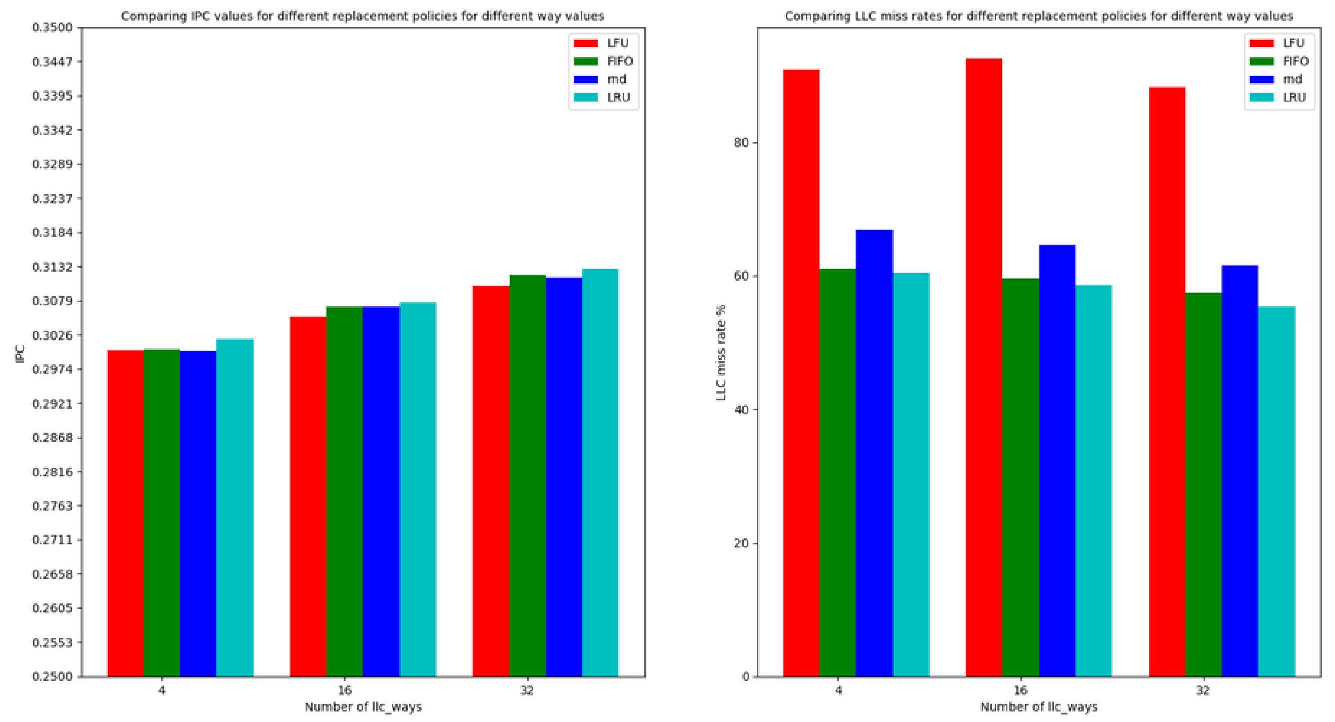
- 🟥 LFU
- 🟩 FIFO
- 🟦 md
- 🟦 LRU

Replacement comparision-INCLUSIVE

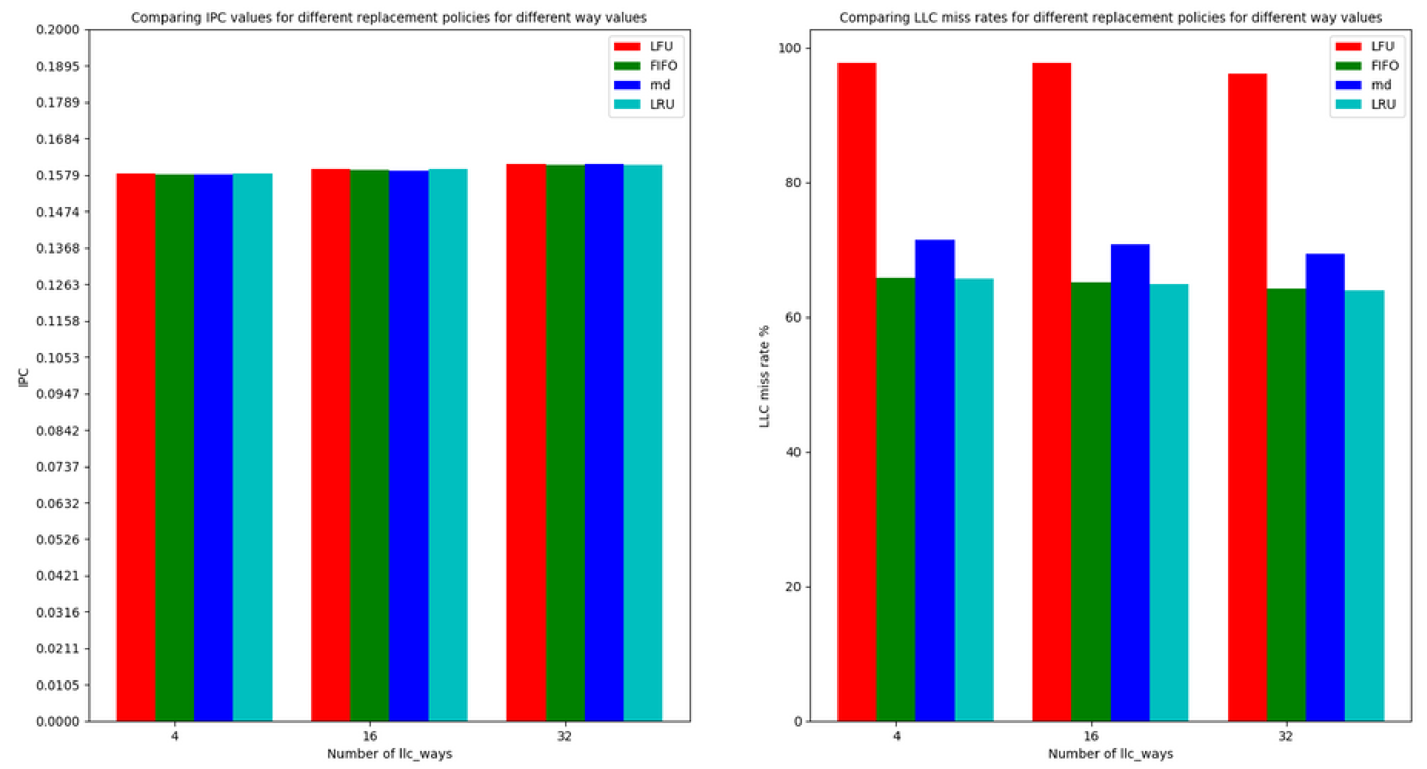
Plot for trace cadical-high-60K-134B.champsimtrace.xz



Plot for trace kissat-inc-high-30K-1802B.champsimtrace.xz



Plot for trace cadical-high-60K-1227B.champsimtrace.xz



Performance(according MISS_RATE): LRU~FIFO>RND>LFU

🟥 LFU
🟩 FIFO
🟦 md
🟦 LRU

REPLACEMENT POLICIES OBSERVATIONS

ACC TO MISS RATE:

SAT Solver traces: Exhibit Temporal locality.

LRU:Effective as it replaces least recently used block which has lower probability to access

FIFO:Accessing block having lower probabilty to access but less effective than lru as not considering recent access history.

Random policy :Less effective than LRU and FIFO because it does not consider the access history of cache lines at all.

LFU :In SAT solver traces, some cache lines may be infrequently used but still important for maintaining temporal locality. Therefore, LFU may evict these useful cache lines and result in higher miss rates.

ORDERS

ACC TO MISS RATE:

LRU~FIFO>RND>LFU

SIZES

- While proportionately changing the sizes of L1I, L1D, L2C, and LLC caches, we observed an increase in IPC values and decrease in miss_rate.
- But this observation is taken on constant latencies.
- Generally IPC values diminish as cache size increases as larger caches may have higher latencies which can offset the benefits of reducing cache misses.

Thank you!

Time to flush our cache and move on to the next task. 😴