

Z-CACHE

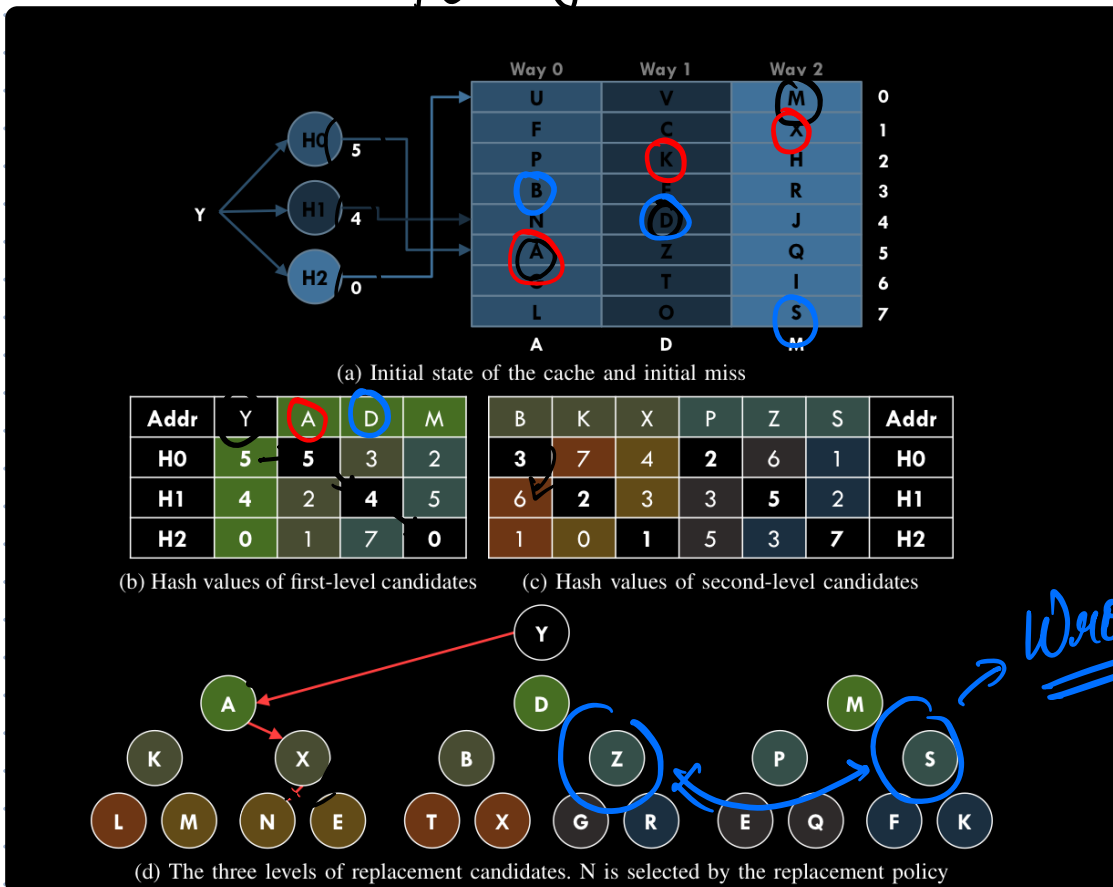
- Decouples associativity and ways
- Associativity is not # of locations that a block can reside in.
It is the number of replacement candidates on eviction

- 1) Walks the tag, to identify replacement candidates
- 2) Pick candidate by replacement policy (LRU) & evict
- 3) Perform a series of relocation to accomodate incoming block

(X) Doesn't increase latency.

Step 1: Walk:


You go in all three places. → create a tree based on the hash of subsequent candidates

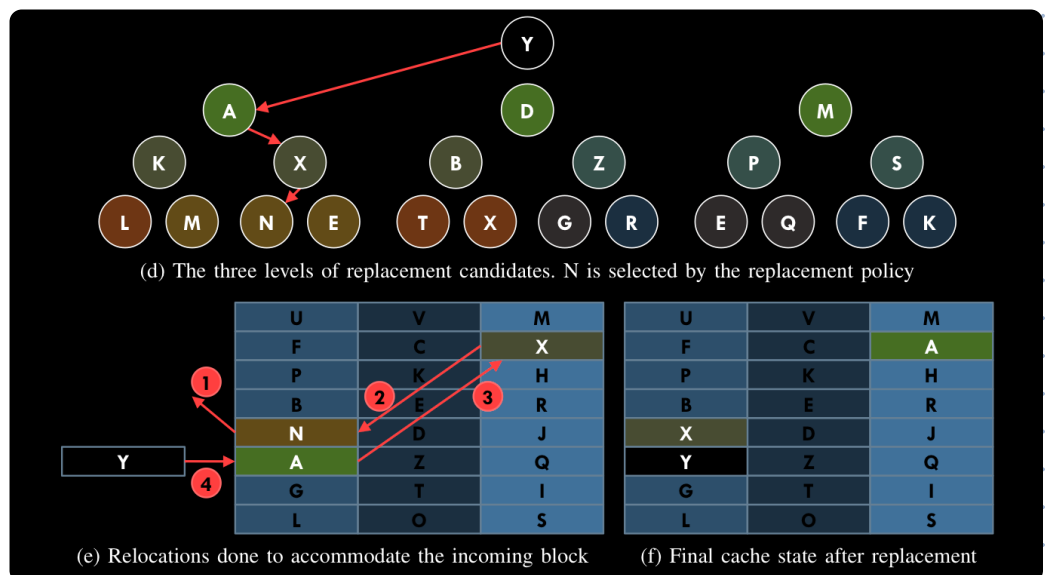


Wrong

- 1) Use hash generator to give possible replacement candidates.
- 2) generate hash for those candidates to get even more candidates.
- 3) go as deep as you want.
- 4) generated number = set number
The generator = Way numbers

Step 2: Relocation

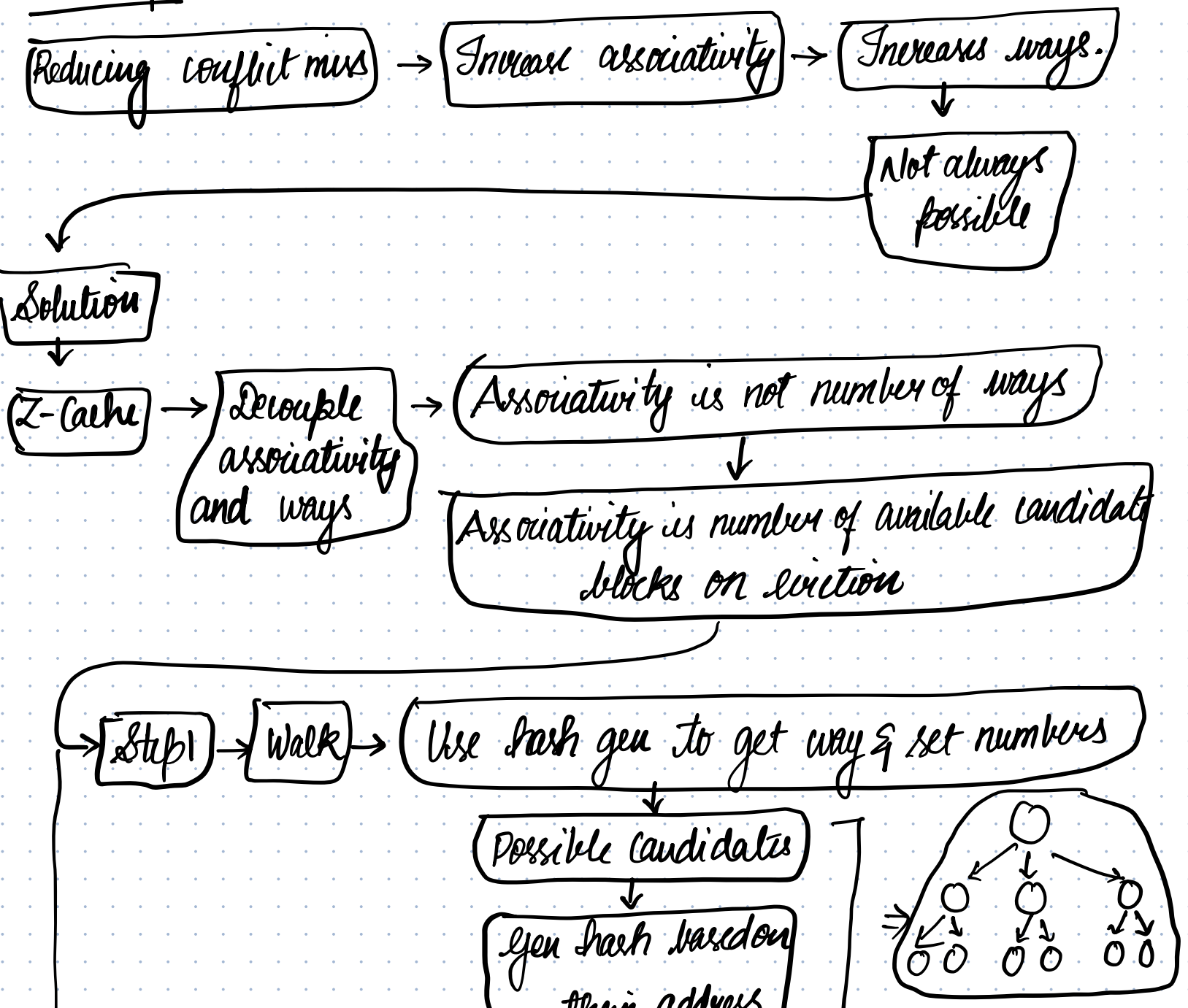
- 1) Once the block to be replaced is chosen, relocate the element in that path and evict it.
- 



Step 3: Replacement Policy:

- Z-cache does not have the concept of set, LRU cannot be used directly.
- global LRU: +1 per access of block
- Bucketed LRU: +1 once every k access.

Mind map:



Insert access



The final level gives all possible candidates

Step 2 → Replacement

Global LRU

+1 counter for access

Bucketed LRU

+1 counter for 1st access.

Step 3 → Relocation

Based on the repl policy select victim



Relocate the ancestors one level down

