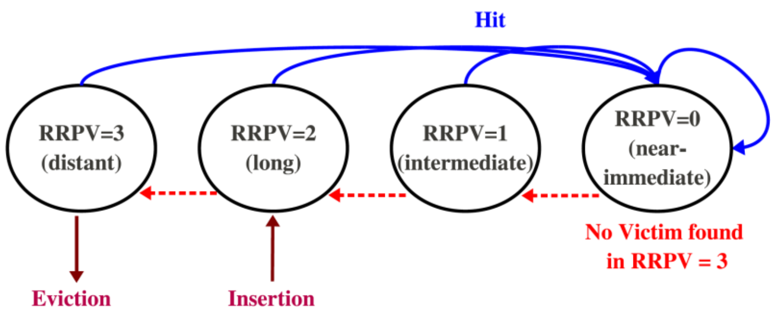
# Introduction to the **Static Re-Reference Interval Prediction (SRRIP)** Cache Replacement Framework

The following text is an excerpt from the paper titled “**High Performance Cache Replacement Using Re-Reference Interval Prediction (RRIP)**,” by A. Jaleel, K.B. Theobald, S.C. Steely Jr., and J. Emer, which was presented at the *2010 International Symposium on Computer Architecture (ISCA)*. The authors propose a cache replacement framework based on the concept of *Re-reference Interval Prediction (RRIP)*:

RRIP uses M-bits per cache block to store one of $2^M$ possible *Re-reference Prediction Values (RRPV)*. …an RRPV of zero implies that a cache block is predicted to be re-referenced in the *near-immediate* future while RRPV of saturation (i.e., $2^M$–1) implies that a cache block is predicted to be re-referenced in the *distant* future. Quantitatively, RRIP predicts that blocks with small RRPVs are re-referenced sooner than blocks with large RRPVs. When M>1, RRIP enables *intermediate* re-reference intervals that are greater than a *near-immediate* re-reference interval but less than a *distant* re-reference interval.

The diagram below shows the so called “**RRIP chain**” with the different RRPV values (M=2 is assumed here, which results in 4 different RRPV values):



(a) RRIP framework

The RRIP chain …represents the order in which blocks are predicted to be re-referenced. The block at the head of the RRIP chain is predicted to have a *near-immediate* re-reference interval while the block at the tail of the RRIP chain is predicted to have a *distant* re-reference interval. A *near-immediate* re-reference interval implies that a cache block will be re-referenced sometime soon while a *distant* re-reference interval implies that a cache block will be re-referenced in the distant future. On a cache miss, the block at the tail of the RRIP chain (i.e., the block predicted to be referenced most far into the future) will be replaced.

…RRIP always inserts new blocks with a *long* re-reference interval. A *long* re-reference interval is defined as an *intermediate* re-reference interval that is skewed towards a *distant* re-reference interval. We use an RRPV of $2^M$–2 to represent a *long* re-reference interval. If the newly inserted cache block has a *near-immediate* re-reference interval, RRIP can then update the re-reference prediction to be shorter than the previous prediction. In effect, RRIP *learns* the block’s re-reference interval.

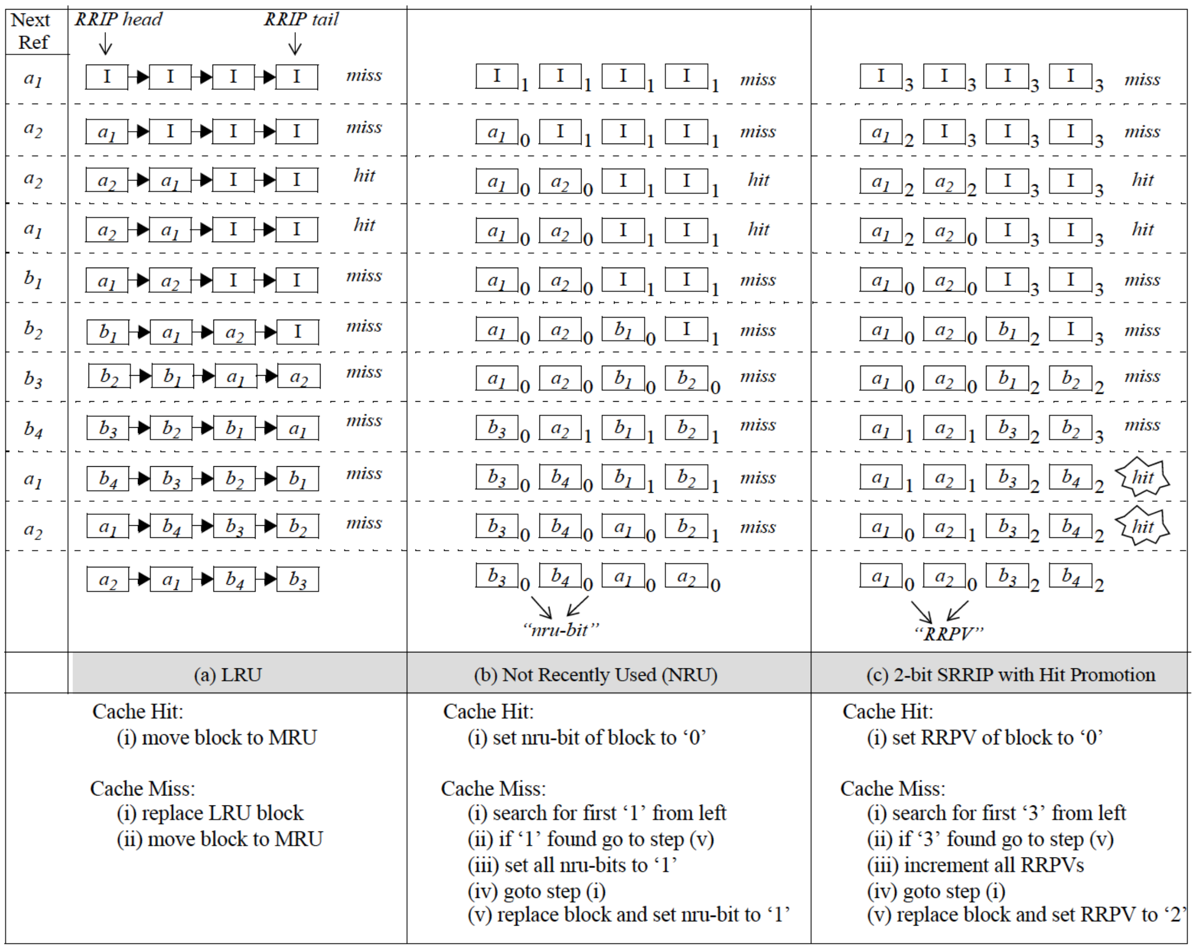
On a cache miss, the RRIP *victim selection policy* selects the victim block by finding the first block that is predicted to be re-referenced in the *distant* future (i.e., the block whose RRPV is $2^M$–1). The victim selection policy breaks ties by always starting the victim search from a fixed location (the left in our studies). In the event that RRIP is unable to find a block with a *distant* re-reference interval, RRIP updates the re-reference predictions by incrementing the RRPVs of all blocks in the cache set and repeats the search until a block with a *distant* re-reference interval is found.

A natural opportunity to change the re-reference prediction of a block occurs on a hit to the block. The algorithm for this update of the RRPV register is called the RRIP *hit promotion policy*. The primary purpose of the hit promotion policy is to dynamically improve the accuracy of the predicted re-reference interval of cache blocks. We propose two policies to update the re-reference prediction: *Hit Priority (HP)* and *Frequency Priority (FP)*.

**Here, we focus only on the HP policy**:

The RRIP-HP policy predicts that the block receiving a hit will be re-referenced in the *near-immediate* future and updates the RRPV of the associated block to zero. The goal of the HP policy is to prioritize replacement of blocks that do not receive cache hits over *any* cache block that receives a hit.

Since the re-reference predictions made by RRIP are statically determined on cache hits and misses, we refer to this replacement policy as *Static Re-reference Interval Prediction (SRRIP)*. **Figure 3c illustrates the behavior of 2-bit SRRIP-HP**. The example shows that SRRIP emulates optimal replacement by correctly predicting a *near-immediate* re-reference interval for the actively used cache blocks…


Next Ref
RRIP head
RRIP tail
miss
hit
"nru-bit"
"RRPV"
(a) LRU
(b) Not Recently Used (NRU)
(c) 2-bit SRRIP with Hit Promotion

| (a) LRU | (b) Not Recently Used (NRU) | (c) 2-bit SRRIP with Hit Promotion |
|---|---|---|
| Cache Hit:<br>(i) move block to MRU<br><br>Cache Miss:<br>(i) replace LRU block<br>(ii) move block to MRU | Cache Hit:<br>(i) set nru-bit of block to '0'<br><br>Cache Miss:<br>(i) search for first '1' from left<br>(ii) if '1' found go to step (v)<br>(iii) set all nru-bits to '1'<br>(iv) goto step (i)<br>(v) replace block and set nru-bit to '1' | Cache Hit:<br>(i) set RRPV of block to '0'<br><br>Cache Miss:<br>(i) search for first '3' from left<br>(ii) if '3' found go to step (v)<br>(iii) increment all RRPVs<br>(iv) goto step (i)<br>(v) replace block and set RRPV to '2' |

**Figure 3: Behavior of LRU, NRU, and SRRIP for a Mixed Access Pattern.**