ECE7995
Caching and Prefetching Techniques in Computer Systems

In-class questions for Presentation:

*Outperforming LRU with an Adaptive Replacement Cache Algorithm*
• What is “scan-resistant” about a replacement algorithm? Is LRU scan-resistant? How about 2Q, LIRS, and ARC? Why?

• Do you agree on the claim that “The DBL(2c) policy is essentially 2Q”? If yes, what are the sizes of $A_{in}(Kin)$, of $A_{out}(Kout)$, and $A_m$ in the context of DBL(2c)?

---

**DBL(2c)**

**INPUT:** The request stream $x_1, x_2, \ldots, x_t, \ldots$

**INITIALIZATION:** Set $\ell_1 = 0$, $\ell_2 = 0$, $L_1 = \emptyset$ and $L_2 = \emptyset$.

For every $t \geq 1$ and any $x_t$, one and only one of the following two cases must occur.

**Case I:** $x_t$ is in $L_1$ or $L_2$.
- A cache hit has occurred. Make $x_t$ the MRU page in $L_2$.

**Case II:** $x_t$ is neither in $L_1$ nor in $L_2$.
- A cache miss has occurred. Now, one and only one of the two cases must occur.
  
  **Case A:** $L_1$ has exactly $c$ pages.
  - Delete the LRU page in $L_1$ to make room for the new page, and make $x_t$ the MRU page in $L_1$.
  **Case B:** $L_1$ has less than $c$ pages.
  1) If the cache is full, that is, $(|L_1| + |L_2|) = 2c$, then delete the LRU page in $L_2$ to make room for the new page.
  2) Insert $x_t$ as the MRU page in $L_1$.  

---


A replacement algorithm is looping resistant if it can produce a hit ratio that is roughly proportional to the ratio between cache size and number of accessed blocks for the looping access pattern (repeated access of a sequence of blocks in the same order). From the following experiment results, we can observe that LIRS is looping resistant while ARC is not? Can you explain ARC is not looping resistant? (If you can explain why LIRS is looping resistant, you get bonus credits.)