LFU-*K*: An Effective Buffer Management Replacement Algorithm

Leonid B. Sokolinsky
Russia
http://www.csu.ru/~sok/e_index.html

Outline

- Related works
- LFU-*K* definition
- Probability model
- Analytic estimation for the *m* parameter
- Heuristic estimation for the *h* parameter
- Experimental Results

Related works

- . . . .
- Robinson J.T., Devarakonda M.V. Data Cache Management Using Frequency-Based Replacement // 1990 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, University of Colorado, Boulder, Colorado, USA, May 22-25, 1990, Proceedings. 1990. P. 134-142.
- O'Neil E.J., O'Neil P.E., Weikum G. The LRU-K Page Replacement Algorithm For Database Disk Buffering // Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26-28, 1993. ACM Press. 1993. P. 297-306.
- Lee D., Choi J., Kim J.-H., Noh S.H., Min S.L., Cho Y., Kim C.-S. On the existence of a spectrum of policies that subsumes the least recently used (LRU) and least frequently used (LFU) policies // SIGMETRICS '99, International Conference on Measurement and Modeling of Computer Systems, May 1-4, 1999, Atlanta, Georgia, USA, Proceedings. 1999. P. 134-143.
- . . . .

Reason

There is no a database buffer management replacement algorithm, which is the best in all cases

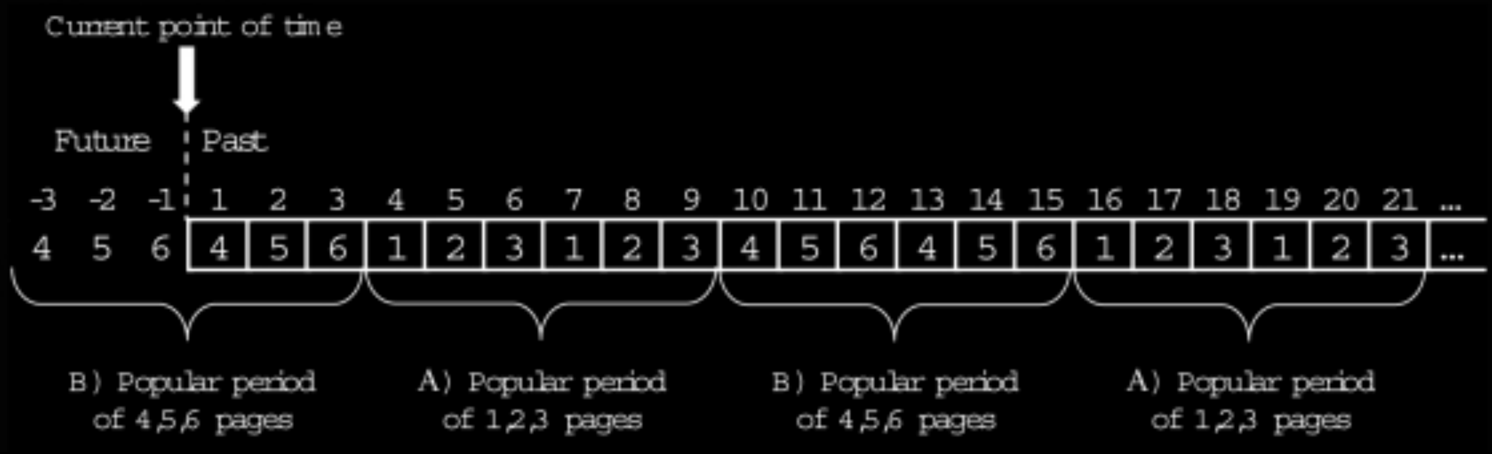
LRU

- LRU replaces the page that was *least recently accessed or used*
- Until the early 80's, the LRU was used almost in all cases
- LRU *fails* in the case of OLTP-style index access

LRU-*K*

- LRU-*K* is proposed by Patrick an Elizabeth O'Neil and Gerhard Weikum
- LRU-*K* is a generalization of LRU (LRU is LRU-1)
- If P_1 page's *K*-th most recent access is more recent then P_2 page's, then LRU-*K* will replace P_1 after P_2
- LRU-2 achieves most of advantage of the LRU-*K* method
- LRU-2 fits the OLTP-style index access
- LRU-2 ***fails*** in the case of *periodical traces*

Periodical traces (simple example)



Position	Buffer pool			
4	1	2	3	4
3	1	2	3	6
2	1	2	3	5
1	1	2	3	4
-1	1	2	3	6
-2	1	2	5	6
-3	1	4	5	6

LRU-2

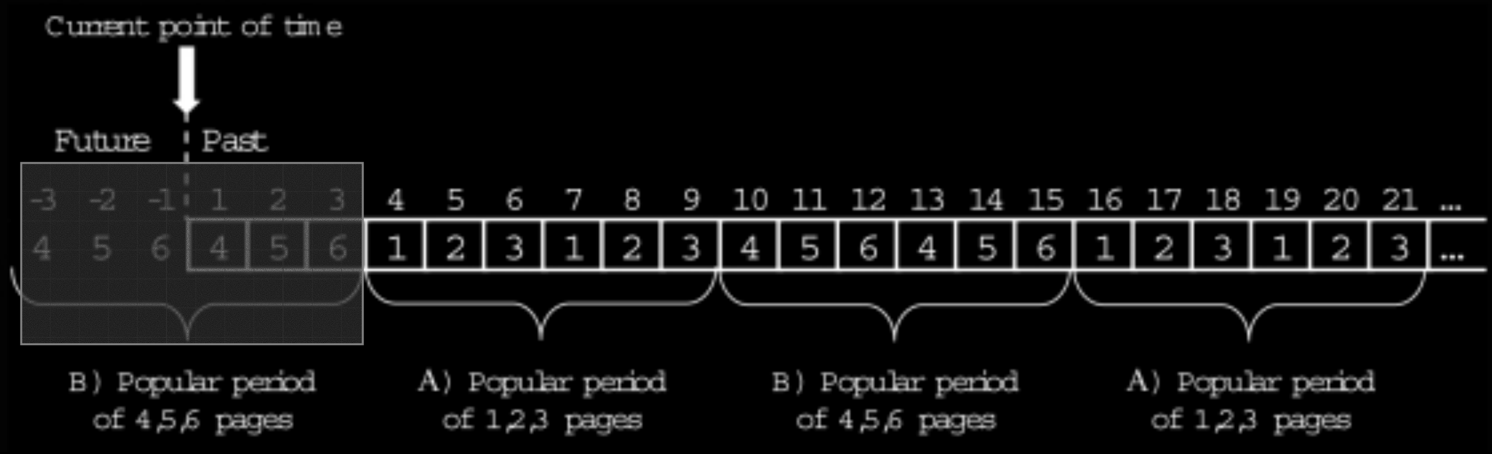
Hit rate: 0%

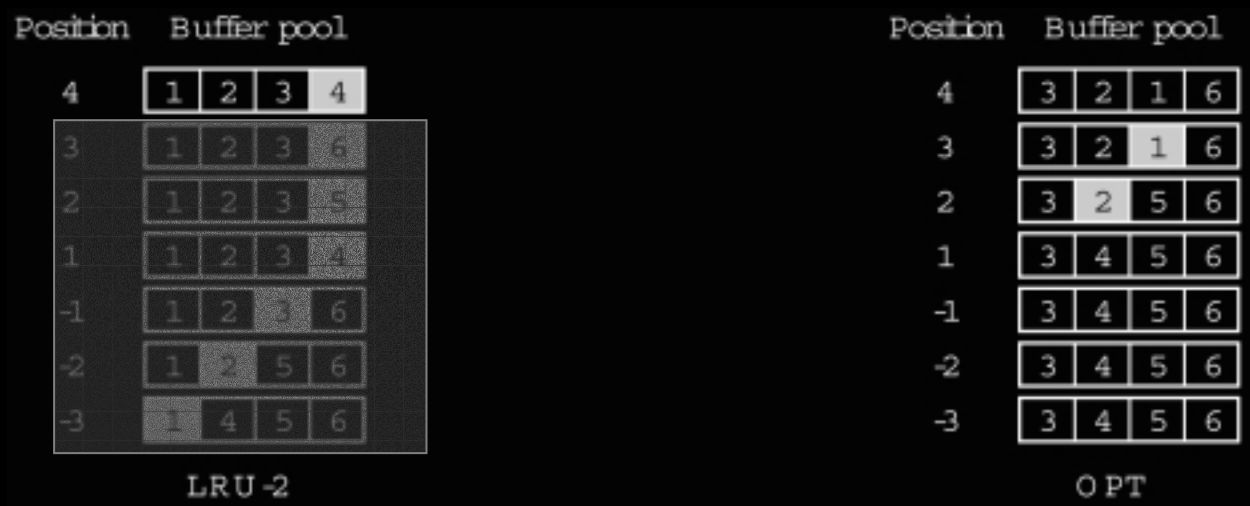
Position	Buffer pool			
4	3	2	1	6
3	3	2	1	6
2	3	2	5	6
1	3	4	5	6
-1	3	4	5	6
-2	3	4	5	6
-3	3	4	5	6

OPT

Hit rate: 67%

Periodical traces (simple example)

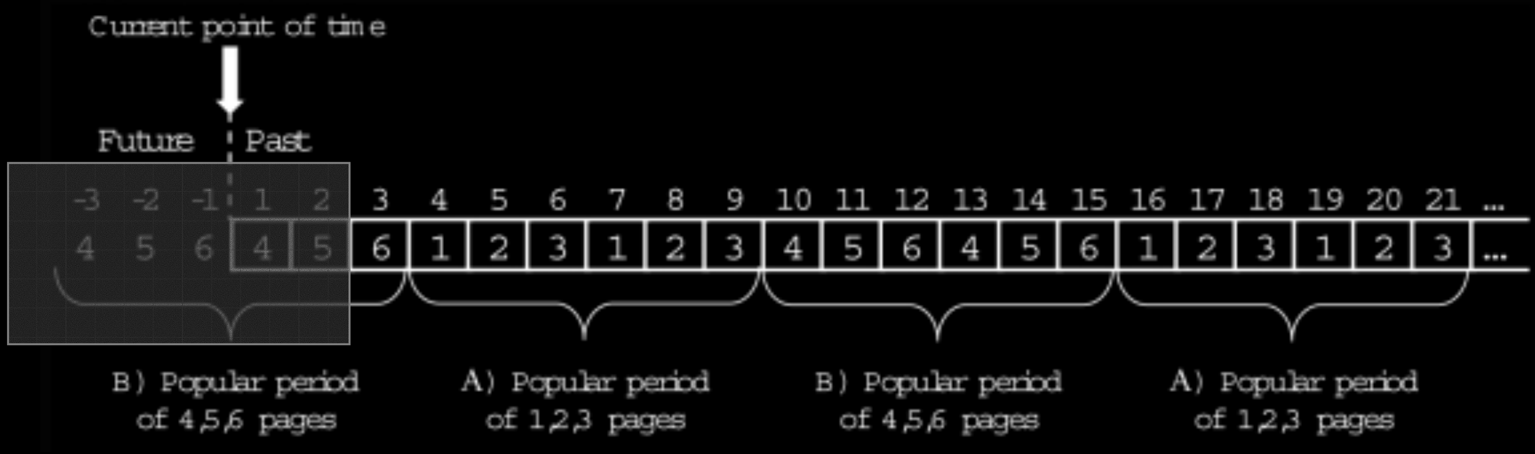


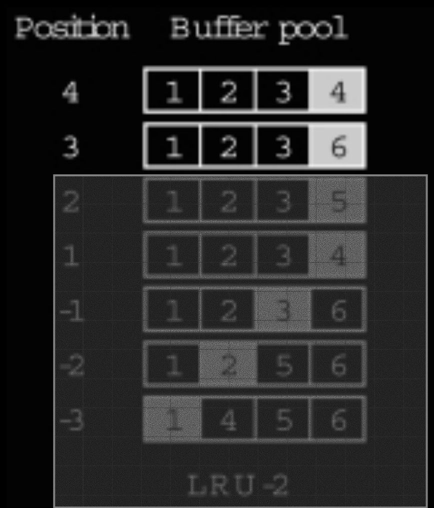


Hit rate: 0%

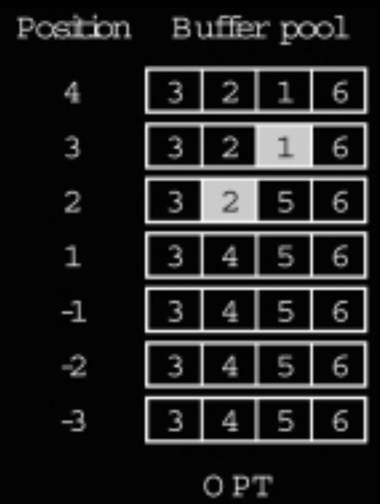
Hit rate: 67%

Periodical traces (simple example)



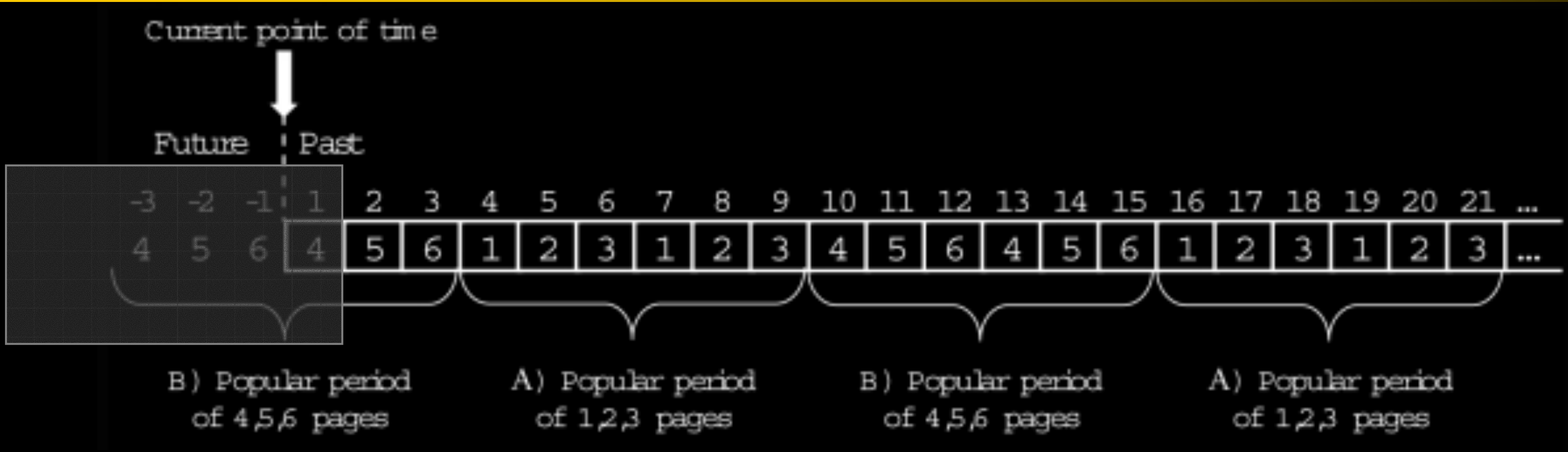


Hit rate: 0%



Hit rate: 67%

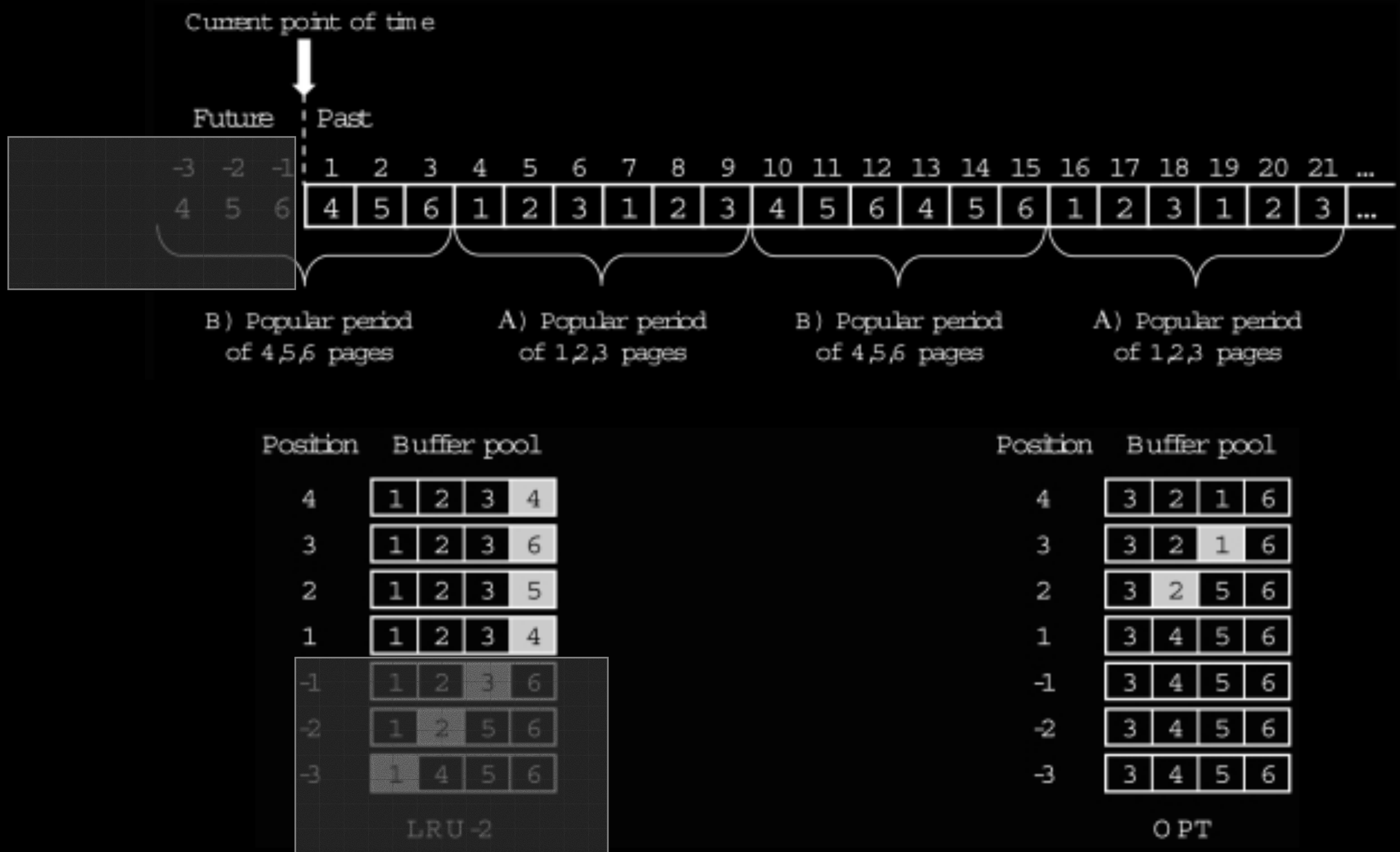
Periodical traces (simple example)



Hit rate: 0%

Hit rate: 67%

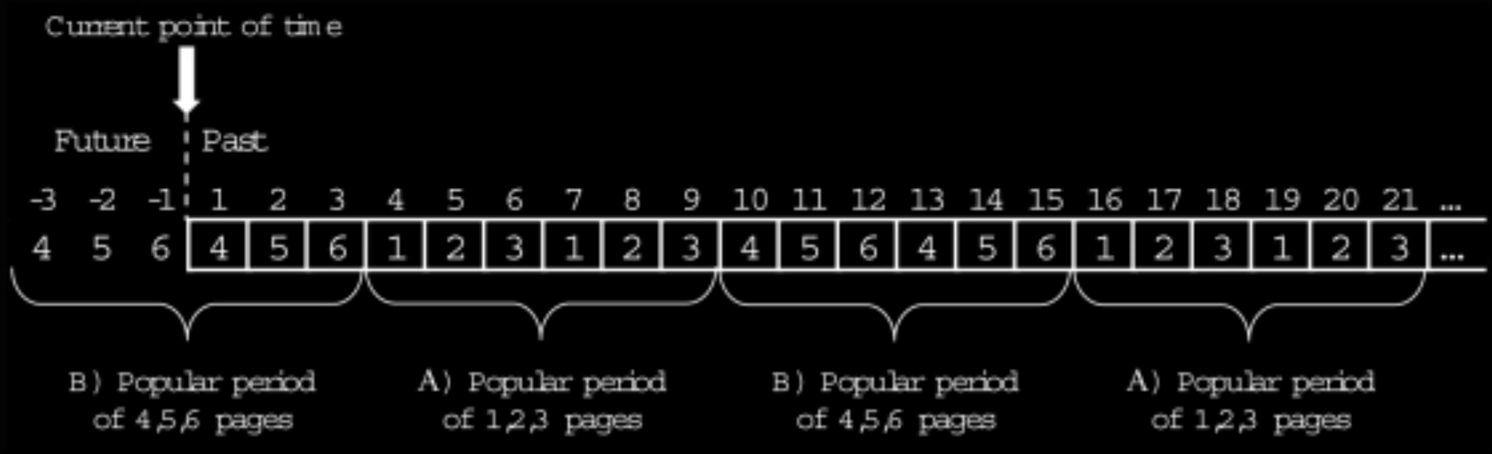
Periodical traces (simple example)

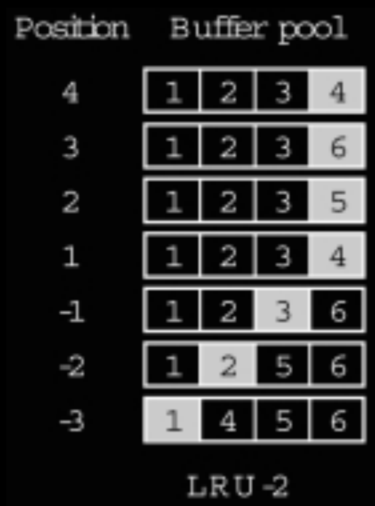


Hit rate: 0%

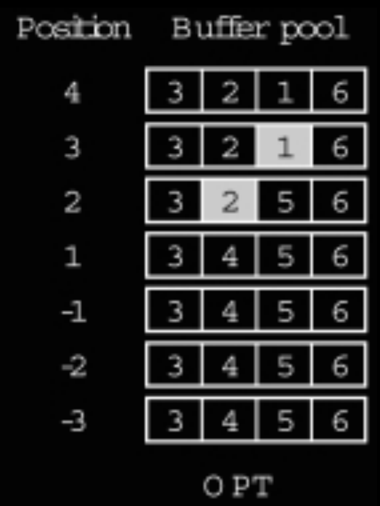
Hit rate: 67%

Periodical traces (simple example)





Hit rate: 0%



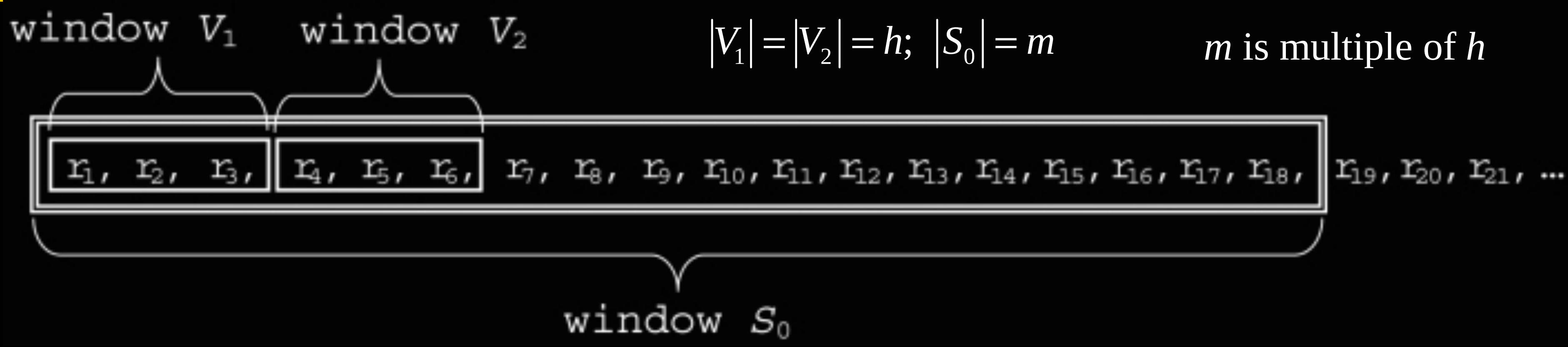
Hit rate: 67%

LFU-*K*

- LFU-*K* is a generalization of LFU that replaces the least frequently used page
- **Informal definition:**
  - **LFU-0** counts amount of references to the page *r* in the reference string segment intercepted by the fixed parameter *m*
  - **LFU-1** takes into account the *"speed"* of growing the amount of references to the page *r*
  - **LFU-2** takes into account the *"acceleration"* of growing the amount of references to the page *r*.
- LFU-2 achieves most of advantage of the LFU-*K* method
- LFU-2 fits the OLTP-style index access
- LFU-2 is suitable for the periodical traces

LFU-2 definition

$|V_1| = |V_2| = h; \; |S_0| = m$ *m* is multiple of *h*



Let: $v_1(r)$ is a function that calculates the amount of references to the page *r* in the window V_1
$v_2(r)$ is a function that calculates the amount of references to the page *r* in the window V_2
$s_0(r)$ - is a function that calculates the amount of references to the page *r* in the window S_0

Define *speed* : $v(r) = v_1(r)$
acceleration: $a(r) = v_1(r) - v_2(r)$
time: $t = m/h$

$$R_{LFU\text{-}2}(r) = s_0(r) + v(r)t + a(r)t^2/2$$

Probability model: distribution law

N – number of caching pages

p_i – reference probability of page i ($1 \le i \le N$)

$$p_i = \frac{1}{i^{\Theta} H_N^{(\Theta)}}$$ – probability distribution law

$H_N^{(s)} = 1^{-s} + 2^{-s} + \ldots + N^{-s}$ – N-th harmonic number of order s

Θ – skew factor ($0 \le \Theta \le 1$)

$\Theta = 0 \iff p_i = \frac{1}{N}$ – uniform distribution

$\Theta = 1 - \frac{\log 0.80}{\log 0.20} \iff$ – "80-20" distribution

$\Theta = 1 \iff p_i = \frac{1/H_N}{i}$ – Zipf's distribution

Probability model: approximate formula

$$H_n^{(s)} = \zeta(s) + \frac{n^{1-s} - 1}{1-s} + O\left(\frac{1}{n^s}\right)$$

$$p_i \approx \frac{1}{i^{\Theta}\left(\zeta(\Theta) + \frac{N^{1-\Theta} - 1}{1-\Theta}\right)}$$

$\zeta(r)$ – Riemann’s zeta function

Analytical estimation for the *m* parameter: "*accurate*" measure

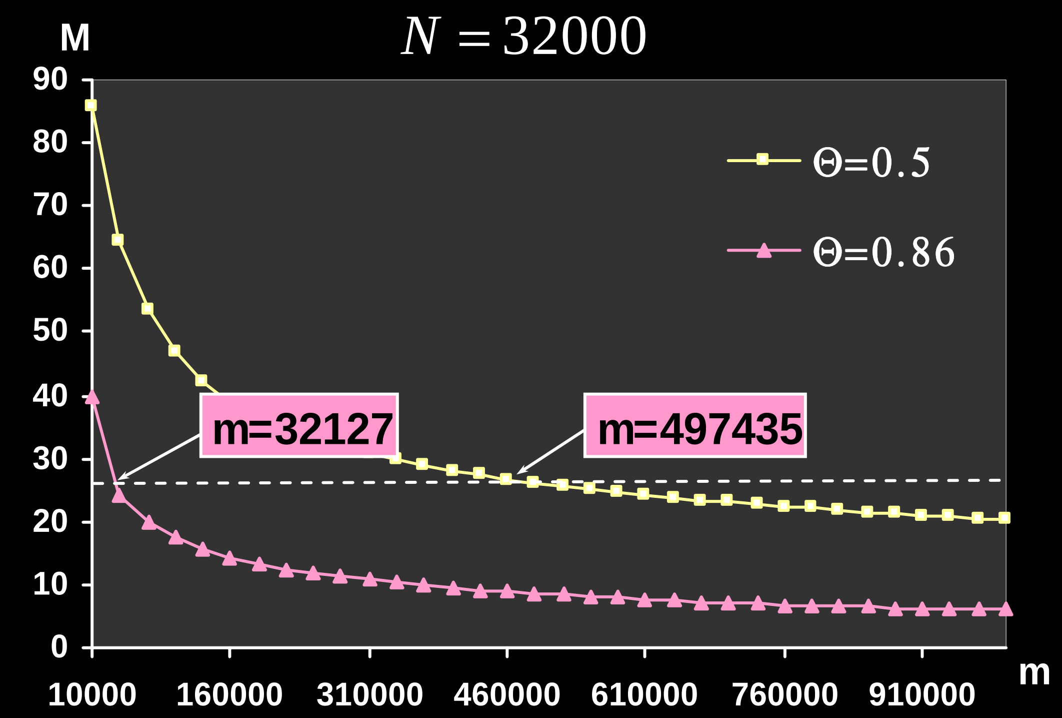
$$\mathbf{M} = \int_0^{1/p_1} v_{1m}(x)\,\mathbf{d}x$$

$$v_{1m}(x) = e^{-(mp_1x+1/2)\ln x+(m(1-p_1x)+1/2)\ln\frac{1-p_1}{1-p_1x}}$$

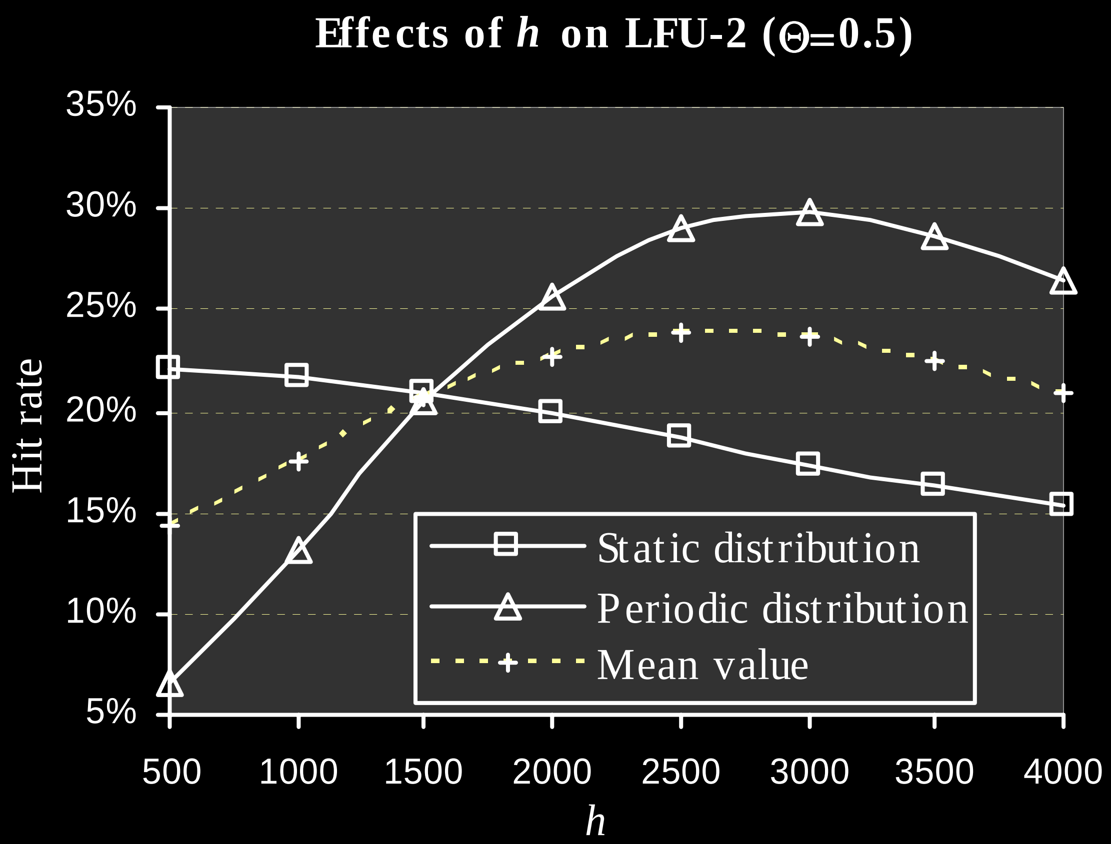
The smaller value of **M** we have, the closer LFU-*K* efficiency to the optimal is achieved.

Analytical estimation for the *m* parameter: "*rough*" measure

$$m > 2\Delta\left(\zeta(\Theta) + \frac{N^{1-\Theta} - 1}{1-\Theta} - 1\right),\ \textbf{where}\ \Delta = 700$$

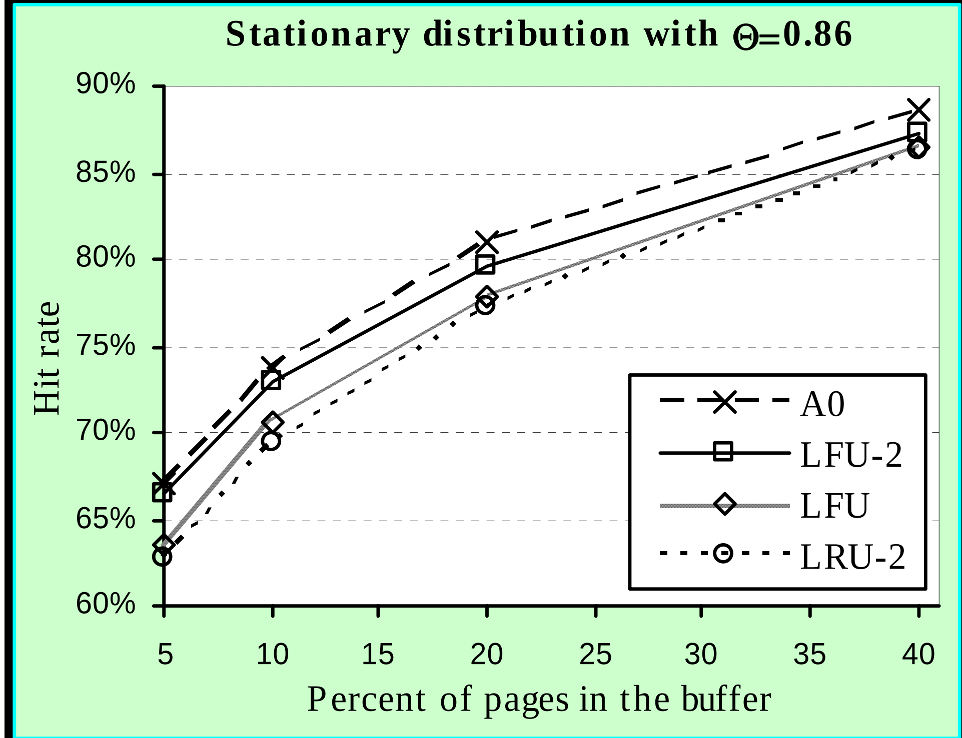


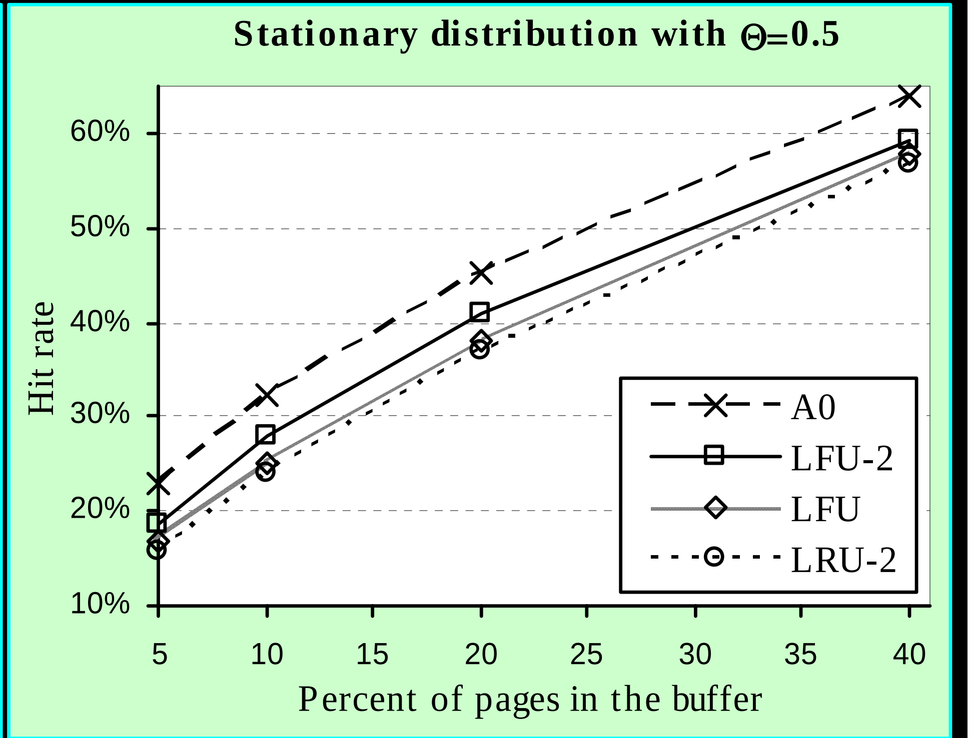
Experimental estimation for the *h* parameter



Effects of *h* on the **LFU-2** algorithm using periodic (with period of 1000) and static distributions with Θ=0.5 and buffer of 1600 pages.

Experimental Results: Stationary Reference Probability Distribution





Experimental Results: Periodic Reference Probability Distribution