Homework Set 1

Exercise 3.5

Page 38

Online Computation And Competitive Analysis

Borodin & El-Yaniv

Statement of Exercise

Prove that CLOCK is *NOT* a marking algorithm.

Notion of the k-phase

Recall any *k*-phase partition of σ contains *k* ***first requests*** for pages:

- ***j*** first requests for pages ***NOT in fast memory*** at start of phase.
- ***k-j*** first requests for pages ***IN fast memory*** at start of phase.
- Any number of 2nd and greater requests for pages in fast memory as phase progresses.

Marking Algorithm Guarantee

Recall also the definition of a marking algorithm:

During any k-phase, any second or greater request for a page is serviced (without fault) from fast memory.

(Restate: Prior to end of phase, no page, once referenced is ever evicted in servicing a faulting request for some other page.)

Algorithm CLOCK

Text lists actions, but what is philosophy:

- Tour cache only to service faults.
- Always tour in fixed sequence.
- After initial fetch, if page is un-requested by next inspection, give it ***one more chance***.
- Thereafter, if un-requested since last inspection, replace.

Is philosophy that of a marking algorithm?

Theorem:

If

(Algorithm CLOCK, *prior to end of a k-phase, evicts at least one page after first request for that page)*

then

(*Algorithm *CLOCK is NOT a marking algorithm.)

Sufficient to Demonstrate:

Algorithm CLOCK, operating with a fast memory of ***k*** pages, on the i^{th} k-phase of σ:

- Services a ***1st request*** $\sigma_{i:[h+0](<k)}$ for a ***page x***,
- ***Evicts page x*** in servicing a 1st request $\sigma_{i:[h+m+0](<k+1)}$ for a different page ***y***, and then
- Later encounters a ***2nd request*** $\sigma_{i:[h+m+r](<k+1)}$ for ***page x*** which must fault.

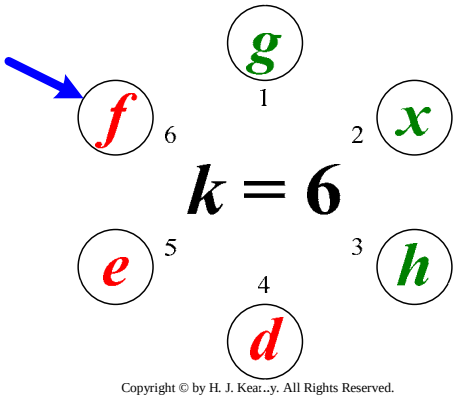
Primary Assumption in Example:

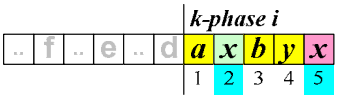
Initial configuration (pages-in-cache, visitation status of each, and current position of clock) is reachable.

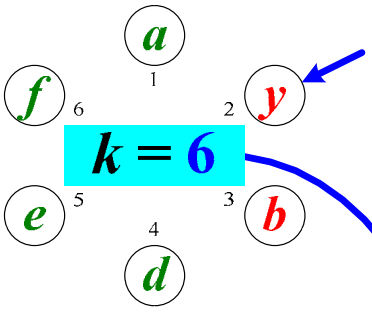
Example:

Initial configuration of fast memory as k-phase i-1 is about to end with the k+1 first request: clock = 6.

Red = bit is SET; Green = bit is CLR.







σ	h	Counter Action		Flag Is	Page & Flag Action	
Last σ_h of k-phase i-1						
a	1	Tally	1	CLR	LOAD: Evict *g*	SET
x	2	-	1	-	-	SET
b	3	Tally	2	SET	-	CLR
		Tally	3	CLR	LOAD: Evict *h*	SET
y	4	Tally	4	SET	-	CLR
		Tally	5	SET	-	CLR
		Tally	6	SET	-	CLR
		Tally	1	SET	-	CLR
		Tally	2	CLR	LOAD: Evict *x*	SET
x	5	*Faults on 2nd request for same page in same k-phase.*				

Therefore:

*This variant of CLOCK (one more chance) is **NOT** a marking algorithm. Q.E.D.*

But, before closing …

What about other variants?

How about a you-specify-the-number-of-chances CLOCK algorithm?

- *Instead of one bit, have n bits and afford from zero to 2^n more chances.*
- *Zero-more-chances fails like one-more-chance, only sooner.*
- Will 2^n *more chances also fail, only later?*

End of Exercise 3.5 Page 38

