Exercise sheet 9.

Data structures — SoSe 2020

László Kozma, Katharina Klost

Due 12:00, June 26th, 2020

Exercise 1 LRU vs FIFO *3+3+1 Points*

Recall the two online paging algorithms, *least recently used* (LRU), and *first-in-first-out* (FIFO). The first algorithm evicts, in case of a cache miss, the page whose last request time is the earliest. The second algorithm evicts the page whose insertion time is the earliest. Suppose the cache size is $K = 3$. You may assume in all cases that the cache is initially empty.

(a) Give an example sequence R_1 where the ratio $LRU(R_1)/FIFO(R_1)$ is arbitrarily close to 3.

(b) Give an example sequence R_2 where the ratio $FIFO(R_2)/LRU(R_2)$ is arbitrarily close to 2.

(c) Can these ratios be larger than 3?

Exercise 2 Least frequently used *3 Points*

Show that the online paging algorithm *least frequently used* (LFU) can not be $f(K)$-competitive for any function f, where K is the cache size. Recall that LFU evicts, in case of a a cache miss, the page that has been requested the fewest number of times since insertion, among those pages that are currently in the cache.

Exercise 3 Changing the cache size *4+0 Points*

Let $LRU(R)$ denote the cost of the LRU algorithm for request sequence R with cache size K, and let $LRU'(R)$ denote the cost of LRU for request sequence R with cache size $K + 1$. In both cases we start with an initially empty cache.

(a) Show that $LRU'(R) \leq LRU(R)$, or in other words, increasing the cache size cannot increase the cost.

(b) Perhaps surprisingly, this is not true for every algorithm, e.g. for FIFO one can construct an example sequence where increasing the cache size *increases* the cost. Study the example in the Wikipedia article "Bélády's anomaly".

(c) Bonus (*+4p*): Check whether this phenomenon can happen for the LFU, LIFO, and LFD algorithms.

Exercise 4 Programming exercise

Please submit the programming exercise report and source code by June 29th.

Total: 14 points. Have fun with the solutions!