Sinem Ozden 150150202

ISE 302 HOMEWORK 4

Q1: Whenever a new page referred it has to be added to memory, so it can be used a page fault occurs. But the referred page may not always be on the memory, so a page fault occurs. To fix this we have to remove one or more pages from cache and replace it with the new page that referred. The best algorithm for page replacement is Optimal Page Replacement algorithm. However, applying Optimal Page Replacement algorithm is impossible. Because to use Optimal Page Replacement it has to be known which page is used mostly. So, I will refer Optimal Page Replacement as a criterion for other algorithms that used in operating systems.

As a page replacement policy Linux uses an algorithm that resembles Least Recently Used Algorithm. In Least Recently Used Algorithm, on the memory the page that hasn't been used or referred for longest time is removed and replaced with the new incoming page. This algorithm is performing almost as good as Optimal Page Replacement in theory. However, we have to track the references of all the pages on the memory which reduces the performance of the algorithm. Because fist we have to traverse all the pages at the memory pick the lowest referred and replace it. So, although Least Recently Used algorithm is really close to Optimal Page Replacement in theory but in practice traversing or keeping the order of the list of pages depends on the implementation reduces the performance.

Windows NT uses clock algorithm instead of least recently used algorithm on uni-processor systems because Least Recent Used requires additional information of how recently each page used. There is a difference between circle algorithm and second change algorithm. The difference is the circle algorithm is a circle linked list instead of a straight one. So, circle algorithm is slightly more efficient than second change algorithm. In some cases, in clock algorithm a reference bit is used to determine if that page used previously. However, for multi-processor systems window choses to use a random page replacement algorithm. Random page replacement is efficient because like circle algorithm the need of tracking references is removed.

For android and iOS rather than performance memory is more important. So, they use either HHD-LRU with batch removal. The difference between Least Recently Used Algorithm is instead removing one page it removes %x of pages that has not been used. It has a slightly lower hit rate than Least Recently Used. Because the cache is not entirely full. Since this algorithm needs to read the metadata of all files in the cache it is not suitable for large caches.

For MAC OS X operating system pageout daemon is used which is similar to First in First out algorithm with smaller differences. Kernel compares number of pages in the free list and a limit value to balance free and inactive list. To replace the pages, when the pages at the free list lower than limit value in free list, pages at the inactive list removed and placed on the free list. Pages at the inactive list can avoid being relaced.

Q2: In Linux Least Recently Used algorithm is used. In windows NT clock algorithm is used.

Least Recently Used algorithm is the closest algorithm to Optimal Replacement algorithm in theory. Because Least Recently Used algorithm tracks the least used page. So, when a new page is referred which is not the memory the algorithm traverses the pages on the memory and choses the page least referred, swaps it out of the memory and replaces it with new incoming page. However, the traversing also consumes a time. Traversing and comparing all the pages expansive and results may not always close to the Optimal Replacement algorithm. There are different ways to implement Least recently used algorithm. The most expensive one is the linked list. The last element of the list is least used page and the first element is the most used page. This implementation is expansive because the list should be updated every time a page is referred even if it was not a new page. The other method requires a hardware support a 64-bit counter. Every time the page is referred counter is incremented. When a new page arrives the page with least value is replaced. The advantage of Least Recently Used algorithm is it never results more than N more page results than Optimal Page Replacement Algorithm. However, to achieve this advantage results in the disadvantage. For N page long list, N+1 pages should be visited. There are different variations of Least Recently Used algorithm to solve this problem.

Clock algorithm is better than both second change and First in First out algorithm, but worse than Least Recently Used algorithm. In First in First out algorithm. The second change algorithm is a modified version of First in First out algorithm and works relatively better. Unlike First in First out algorithm instead of immediately removing it first checks the reference bit. If reference bit is not set then page if swapped out and replaced with new referred page. If reference bit is already set than reference bit is cleared and inserted at the back of the que like a new referred page. The difference between second change algorithm and clock algorithm unlike second change algorithm, instead of a linear list clock algorithm uses a circular list. A pointer is used to point last page that referred. When a page fault occurs referenced bits checked starting form pointer. There are different versions of clock algorithm. The Clock with Adaptive Replacement works better than both Least Recently Used Algorithm and Clock Algorithm.

To conclude it is hard to suggest one algorithm is superior to other. For example, page fault of Least Recently Used Algorithm is lower than Clock algorithm because it tracks which referred how many times and choose to which page to be replaced. However, because Least Recently Used algorithm tracks which page referred how many times and make a selection according to that information clock algorithm performs better. So, it is better to adopt a page replacement algorithm according to the situation.

REFERENCES:

https://en.wikipedia.org/wiki/Page_replacement_algorithm#Techniques_for_hardware_with_no_reference_bit

https://www.nada.kth.se/utbildning/grukth/exjobb/rapportlistor/2012/rapporter12/hultgren_daniel_12073.pdf

https://www.google.com.tr/url?sa=t&rct=j&q=&esrc=s&source=web&cd=11&cad=rja&uact=8&ved=2ahUKEwiOz_aOz8ffAhUGDuwKHfkyDgIQFjAKegQIBBAC&url=https%3A%2F%2Fusers.cs.jmu.edu%2Fabzugcx%2FPublic%2FStudent-Produced-Term-Projects%2FOperating-Systems-2003-FALL%2FMacOSX-by-Tomomi-Kotera-2003-Fall.doc&usg=AOvVaw2KYfctvBpEfdXaS-9d6yD7

https://www.kernel.org/doc/gorman/pdf/understand.pdf

https://www.itprotoday.com/compute-engines/inside-memory-management-part-2

https://www.kernel.org/doc/gorman/html/understand/understand013.html