Plagiarism Checker X Originality Report

Similarity Found: 11%

Date: Thursday, October 22, 2020
Statistics: 252 words Plagiarized / 2257 Total words
Remarks: Low Plagiarism Detected - Your Document needs Optional Improvement.

Analysis of Random Page Replacement Algorithm in Operating System Rakhmat Purnomo3, Tri Dharma Putra1, Wowon Priatna2 Department of Informatics, Faculty of Computer Science, University of Bhayangkara Jakarta Raya Jalan Perjuangan Bekasi Utara, West Java, Indonesia1,2,3 Abstract: Page replacement algorithms choose pages to swap out from memory, namely when a new page needs allocation memory. Several page replacement algorithms proposed by experts.

Four well known page replacement algorithms are Least Recently Used (LRU), Optimal Page Replacement (OPR), First In First Out (FIFO), Random page replacement algorithms. Each one has the objective to minimize the page fault. If the minimum page fault achieved, the performance of the process increased. The algorithm discussed is random page replacement algorithm. In this paper three case studies of random page replacement algorithm are discussed.

Keywords: Random page replacement algorithm, page fault, page replacement algorithm, FIFO, LRU, OPR

Introduction Memory management is the most important part of operating system in which all activities related to the memory are perform [1]. The replacement of page is the main concept. Pages is required to be in the main memory [2]. Computer system objective is to execute program of diferent sizes.

The program should be in main memory when program is executed. But main memory has limited size to accommodate programs which is large. Execution of program that may not be entirely in main memory is called virtual memory [3]. Virtual memory paved the way to execute program that is larger than main memory.

The process in which we provisionally moves process from primary memory to the hard disk or seceondary memory, so that the memory available for other processes, which is known as swapping [4] On operating systems which use paging for memory management, it is needed a page replacement algorithm to decide which pages to replace when a new page is entered. Good replacement can reduce the cost of page fault that result in higher performance of the system [5].

Page Fault occurs when a process references to a page that is not present in the main memory dan needs to be through from secondary memory [6]. when a program currently executing accesses a memory page that is mapped to the virtual address space, but is not loaded into physical memory. Because actual physical memory is much smaller than virtual memory, a page fault occurs.

In this case, the operating system may have to replace one of the existing pages with the new page it needs. There are several page replacement algorithms for this replacement. One of them is the random replacement algorithm [7]. Some Wellknown Page Replacement Algorithms There are several other algorithms of page replacement proposed by experts, namely : First in First out (FiFO): This is the simplest page changing algorithm. In this algorithm, the operating system keeps track of all the pages in memory that are in the queue.

Page has been in front of the queue the longest. When a page has to be replaced the page at the front of the queue is chosen to be deleted. An average of hit ratio of 50 percent is reported by FIFO algorithm. The advantages and disadvantages of FIFO algorithm: Easy to understand, very easy to be implemented, Performance not always good [8].

Optimal Page Replacement: In this algorithm, pages are replaced which will not be used for the next longest duration of time. Optimal page replacement is a good algorithm, but it is impossible to implement practically, because the operating system will not

know the next request. The use of this algorithm is to create benchmarks so that other replacement algorithms can be analyzed against this algorithm.

On this algorithm, system select victim whole reference is furthest. This algorithm has the best page fault behavior reported . Least Recently Used: In this algorithm, the page will be replaced the least used (least recently used). An average of 60 % hit ratio is reported by LRU algorithm. This algorithm selects the victim page as that which is not been demanded for access for along time.

Random Page Replacement Algorithm: In this algorithm, the operating system will keep track of all the pages in memory which are marked as the next replaced slot. And the selection of which slots to replace is done randomly. Parameter of The Algorithms Page replacement algorithm depends on the following parameters: Page fault rate, indicates by how many miss occur for asset of page references.

The minimal the page fault rate, the higher the efficiency of the algorithm. Hit rate, indicates how the hit occurs on the page. The higher hit rate, the higher the efficiency of the algorithm [9]. Random Page Replacement Algorithm In page replacement using a random algorithm (random) every time a page fault occurs, the replaced page is randomly selected.

This technique does not use any information in determining which pages to replace, all pages in main memory have the same weight to be selected. How to select random or arbitrary pages, including the page that is currently referenced (pages that should not be replaced). This random technique is very bad, the experiment shows the random algorithm causes a very high frequency of page faults .

Case Study As like its name, in this random page replacement algorithm, page replacement is done randomly. The page that refers to slot randomly, its slot place will be replaced by the new coming page on the queue. In each case study, we use just three frames. Here the random slot will be given a asterisk (*) sign. 5.1 Case Study 1 Let us discuss case study 1, as we can see from the table below: TABLE 1.

Case Study 1 Page queue _0 _1 _3 _4 _ _2 _0 _0 _4 _ _1 _3 _1 _3 _ _Random Replacement _0 _0 _0* _4 _ _4* _0* _^ _4 _ _4 _4* _1* _^ _ _ _ _1 _1 _1* _ _2 _2 _^ _2 _ _2 _2 _2 _^ _ _ _ _ _3 _3 _ _3 _3 _^ _3* _ _1* _3 _3 _^ _ _Page Fault _x _x _x _x _ _x _x _ _x _ _x _x _x _ _ _ Let us discuss the first string. First 0 gets in, page fault happens. Then 1 gets in, page fault happens. Then 3 gets into the slot, still page fault happens.

The random slot to be filled in is the first frame, namely 0. This slot is chosen randomly.

And in the next page, namely 4, will fill in the slot of 0, which is pointed by the asterisk sign. Still in this analysis, the page fault happens. And then 1 gets chosen as the random slot. The second string, 1 is the random slot chosen.

2 gets in into that slot, still page fault happens, since there is no hit. 4 gets chosen randomly tobe filled in. Then 0 gets into the first frame, replacing 4. This is also a page fault, it is a miss. Then 0 get asterisk to be the next chosen slot. Then the next page, the 0, gets a hit, since page 0 is in the first frame. There is no page fault. This is a hit. Then, 4 gets into the first frame, replacing 0, page fault happens. Then 3 is pointed by asterisk. The last string.

Since the third frame has the asterisk, which is get the random priority, 1 will get into third frame replacing 3. Then the first frame, 4, gets the asterisk randomly. The next one, 3 replacing 4 and this first frame gets chosen as the random slot. Page fault happens. The last one is 3. In this last page replacement, 3 gets a hit. There are ten misses in this algorithm and there are two hits. 5.2. Case Study 2 Let us discuss case study 2, as we can see from the table below : TABLE 2.

Case Study 2 Page queue _0 _1 _3 _2 _ _1 _4 _3 _2 _ _1 _2 _4 _4 _ _Random Replacement _0 _0 _0 _0 _ _0 _0 _0 _0 _ _0 _^ _0 _^ _ _ _ _1 _1* _2 _ _2 _2* _3 _3* _ _1 _^ _1 _^ _ _ _ _ _3 _3* _ _1* _4 _4* _2 _ _2* _^ _4* _^ _ _Page Fault _ x _x _x _x _ _x _x _x _x _ _x _ _x _ _ _ Take a look at the first string. 0 gets in, this is page fault. Then 1 gets in, this is also a miss, a page fault. Then 3 gets in into the third slot, this is also a page fault.

In the meantime, 1 gets priority randomly. Then 2 replace 1. This is also a page fault. Then 3 gets asterisk. The second string, 1 gets into the third slot, this is a miss, a page fault. Also 1 get asterisk. Then 4 replaces 1, this is also a page fault. 2 gets asterisk. Then 3 replaces 2, this also a miss, a page fault, with 4 gets asterisk randomly.

Then 2 gets into into third slot, replacing 4, with 3 gets asterisk as the next priority. The last string, page 1 replacing page 3, because page 2 has asterisk which means should be replaced. Then, the second page, 2, gets a hit in the third slot. Then the third page, 4, replacing page 2, with page 4 itself gets asterisk. The last page also gets a hit in the third slot, page 4.

There are ten misses and two hits happen. 5.3. Case Study 3 Let us discuss case study 3, as we can see from the table below: TABLE 3. Case Study 3 Page queue _0 _2 _3 _2 _ _1 _0 _2 _2 _ _4 _1 _0 _2 _ _Random Replacement _0 _0 _0 _^ _ _0* _^ _2* _^ _ _4 _^ _4* _2 _ _ _ _2 _2* _^ _ _1 _^ _1 _^ _ _1* _^ _0 _0 _ _ _ _ _3 _^ _ _3 _^ _3 _^ _ _3 _^ _3 _3 _ _Page Fault _x _x _x _ _ _x _ _x _ _ _ x _ _x _x _ _ Take a look at the first string.

0 gets in this is a page fault. Then 2 gets into the second slot, also a page fault happens. Then 3 gets into the third slot, 2 gets asterisk randomly. Then page 2 gets a hit. The second string, 1 replacing 2, this is a page fault, and 0 get asterisk randomly. Then 0 gets a hit in the first slot. Then 2 replaces 0, and the 2 itself get asterisk randomly. Then the last page, 2, gets a hit.

The last string, 4 gets into the first slot, replacing 2, and then 1 gets pointed by asterisk randomly. Then the second page, 1 gets a hit. Then the third page, 0, replacing 1, this is a page fault. And 4 get pointed by asterisk randomly. The last page, 2, replacing 4, this is also a page fault. There are eight page fault and four hits happen. Conclusion From the three case studes, the first case study we get ten misses and two hits.

For the second case study, we get ten misses and two hits. And from the third case study, we get eight page faults and four hits. The third case study is more efficient then the two case studies before, because the third case study get four hits and eight misses, in the meantime, the first study case studies only get two hits and ten misses each. The conclusion is the third case study is more efficient then first and second case study.

It is concluded that this algorithm (random page replacement algorithm) is not so efficient from other wellknown page replacement algorithms. In the first case study, we have ten misses and two hits, so we get the efficiency is 16,7 percent, in the meantime the third case study we have eight misses and four hits, so we get efficiency 33,3 percent. References

[1] N. H. Ali A. Titinchi, “A New Method to Enhance LRU Page Replacement Algorithm Performance,” Int. J.

Sci. Technol. Res., vol. 9, no. 2, pp. 4185–4190, 2020. [2] G. Rexha, E. Elmazi, and I. Tafa, “A Comparison of Three Page Replacement Algorithms: FIFO, LRU and Optimal,” Acad. J. Interdiscip. Stud., 2015, doi: 10.5901/ajis.2015.v4n2s2p56. [3] S. Shastri, A. Sharma, and V. Mansotra, “Study of Page Replacement Algorithms and their analysis with C#,” 2016. [4] G. P. Arya, D. Prasad, and S. S.

Rana, “An improved page replacement algorithm using block retrieval of pages,” Int. J. Eng. Technol., 2018, doi: 10.14419/ijet.v7i4.5.20004. [5] B. A. T. V. L. Kolhe, “Analysis of Various Page Replacement Algorithms in Operating System,” Int. J. Sci. Res., 2016. [6] A. Saxena, “A Study of Page Replacement Algorithms,” Int. J. Eng. Res. Gen. Sci., vol. 2, no. 4, pp. 385–388, 2014. [7] G. for Geeks, “Geeks for Geeks,” Page Replacement Algorithm, 2020. https://www.geeksforgeeks.org/page-replacement-algorithms-in-operating-systems/ (accessed Oct. 08, 2020). [8] D. P. Juhi Kumari, Sonam Kumari, “A Comparison of Page Replacement Algorithms: A Survey,” Int. J. Sci. Eng. Res., vol. 7, no. 12, pp.

57–62, 2016. [9] R. R. Hitha Paulson, “Page Replacement Algorithms - Challenges and Trends,” Int. J. Comput. Math. Sci., vol. 6, no. 9, pp. 112–116, 2017. BIOGRAPHY <Optional> Authors have the option to publish a biography together with the paper, with the academic qualification, past and present positions, research interests, awards, etc. This increases the profile of the authors and is well received by international reader.

<1% - https://technology.blurtit.com/3545809/what-is-the-purpose-of-main-memory
<1% -
https://afteracademy.com/blog/what-is-virtual-memory-and-how-is-it-implemented
1% -
https://www.researchgate.net/publication/261836721_An_Effective_Round_Robin_Algorithm_using_Min-Max_Dispersion_Measure
3% -
https://www.geeksforgeeks.org/page-replacement-algorithms-in-operating-systems/
<1% - https://study.com/academy/lesson/page-replacement-definition-algorithms.html
1% - https://www.youtube.com/watch?v=QbbOb8SvAPk
<1% -
https://www.researchgate.net/publication/338026849_A_page_replacement_algorithm_based_on_a_fuzzy_approach_to_improve_cache_memory_performance
1% - https://technobyte.org/page-replacement-algorithms-in-os/
1% - https://en.wikipedia.org/wiki/Page_replacement_algorithm
<1% -
https://www.ijser.org/researchpaper/A-Comparison-of-Page-Replacement-Algorithms-A-Survey.pdf
<1% -
https://www.slideshare.net/vtunotesbysree/solution-manual-of-operating-system-concepts-by-abraham-silberschatz-peter-baer-galvin-greg-gagne
<1% -
https://www.researchgate.net/publication/319997969_Page_Replacement_Algorithms_-_Challenges_and_Trends
<1% -
http://www.expertsmind.com/questions/explain-the-different-page-replacement-policies-30162918.aspx
<1% -
http://the-eye.eu/public/Books/robot.bolink.org/Simulation%20of%20Page%20Replacement%20Algorithms.pdf
<1% -
https://stackoverflow.com/questions/6398811/can-a-tlb-hit-lead-to-page-fault-in-memory
<1% -
https://www.researchgate.net/publication/220788195_LFU-K_An_Effective_Buffer_Management_Replacement_Algorithm
<1% - http://www.theijes.com/Vol,5,Issue,1.html
1% -
https://www.researchgate.net/publication/327964913_An_Improved_Page_Replacement_Algorithm_Using_Block_Retrieval_of_Pages

1% - https://www.medsci.org/ms/author