International Journal of Scientific Research and Engineering Development-– Volume 2 Issue 6, Nov- Dec 2019

Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

HYBRID PAGE REPLACEMENT ALGORITHM

Egbunu Charity O.,[1]Ogedengbe Matthew T.,[2]Ollo Vincent.[3]

[1]*Departmentof Mathematics/Statistics/Computer Science, University of Agriculture Makurdi, Nigeria.*

[2]*Departmentof Mathematics/Statistics/Computer Science, University of Agriculture Makurdi, Nigeria.*

[3]*Departmentof Mathematics/Statistics/Computer Science, University of Agriculture Makurdi, Nigeria.*

[1]charityakowe@gmail.com[2]matt554real@gmail.com

[3]ollovincent@gmail.com

ABSTRACT

Page replacement algorithms play essential roles in virtual memory management, especially in Operating Systems that uses paging. Pagereplacement occurs when a requested page is not found in the memory (page fault) or a free available page is not enough to satisfy the request. This is because there are none, or the number of free pages is lower than the number requested. In this study, two standard page replacement (Least Recently Used LRU and OptimalOPT algorithms) were hybridized as hybrid page replacement (HRA). HRA based on the concept of OPT and LRU for it operation. The HRA was evaluatedby comparingit performance in terms of number of page faults generated with the standard page replacement algorithms, namely First InFirst Out FIFO, LRU and Optimal. Results showed that HRA outperformedFIFO, OPT and LRU when the number of frames increases to 4 and above.

Keywords - Memory, Frame, Page fault,Paging.

INTRODUCTION

Page replacement process is paramount in operating system and page replacement algorithms are responsible for the decision on which page should be removed from the memory when a requested frame is not available. However, some algorithms have been developed to solve this problem. The algorithms differ from each other in the way in which the page removal process is been handled (Genta, 2015). All page replacement algorithms are internally similar to the following: insertion, detection and searching for a page. However, most of these algorithms depends on specialized data structure for effective performance (Kavaraid, 2013). Most of the algorithms handle huge amount of data, the locality of reference is considered as a shared attribute between programs. Hence, the majority of applications does not access all data at the same time. Instead of this, they reference only small part of data at different point in time (Shen et al, 2004). The locality of reference can be adopted in two different approaches: spatial locality of reference and temporal locality of references. The spatial locality is of the view that nearby memory location will be referenced in the near future while temporal locality depends on the time the page spends in the frame (O'neil*et al.,* 1993).

The localities of reference can be increased by careful selection of data structure used in the algorithm. The data structure will reduce the page fault rate as well as the number of pages in working set. For example, a stack has high locality because the replacement algorithm of the stack always accesses the top. Some measures: memory reference, search speed and a total number of page touched are used to measure the performance of page replacement algorithms(Khulbe et al, 2014). The efficiency of page replacement algorithm is an open

question for the study since there are several factors that affects the performance of an algorithm (Brinkhoff, 2002). The memory access has a clear impart on page replacement algorithm performance (Anthony, 2015).

To analyze the performance of page memory system, some page frame number (PFNs) has to be generated during the execution of a given program (Hwang and Jotwani, 2011). Several properties should cover the effective replacement algorithm, it should be able to distinguish the hot and cold resident in cache to reduce the number of page faults. Furthermore, efficient algorithm implementation can be achieved if it uses a constant and small portion of the memory that stores page history in the cache, it should not consume large memory space. Generating minimum number of page fault implies better performance and this study proposed the hybrid page replacement algorithmwhich is a combination of the OPT and LRU page replacement algorithms. The performance of the hybrid system will be compared with the existing page (FIFO, LRU and OPT) replacement algorithms.

II. Related Works

Park, *et al.*,(1986), worked on Clean First Least Recently Used (CFLRU). It was the first page replacement algorithm ever proposed for the NAND flash memory. This was initially designed in consequence of the unique hardware constraints posed by the NAND flash storage memory. The algorithm was based on the original LRU algorithm and was enhanced to fit the needs of the operating system. The concept of cold and hot pages was introduced on the existing Clean and Hot pages.The CFLRU considered the entrance recurrence of every page inside of the cold-first regions of the clean page rundown and dirty page list. The performance of CFLRU algorithm was compared with those of LRU and the CFLRU algorithm demonstrated the most noteworthy page hit ratio.Jung, *et al.*, (2000), proposed Least Recently Used-Write-Sequence-Reordering(LRU-WSR).The proposed LRU-WSR was a modified version of LRU introduced with Write Sequence Reordering (WSR) strategy. The LRU-WSR algorithm maintains a single page list in the LRU order similar to the original LRU algorithm. However, any dirty pages in the list will have a cold flag which may or may not be set as per the situational operation. The identification of Cold or Hot of a dirty page is taken care of by a cold detection algorithm. A dirty page may be cold, if that is the least referenced page in the list and its cold flag is set. A dirty page may be hot if that page is referenced again and the cold flag on it is cleared. Results from this work was compared with FIFO and found 18% better off than FIFO.Mark Page Discard Randomly Caching Algorithm (MPDRC) was proposed by Sumit*et al.*, (2015), the work was developed in windows XP using C# .Net and the unique identification number was assigned to unique URL‘s to log of proxy server. These numbers were taken as a reference string that suits input to the innovative algorithms. The focus of the research was exploration of web proxy caching algorithm best for proxy server. With the help of log in details of proxy server, a real trace web reference was obtained. For the simulation, each URL was given a numeric identity which enables the numeric reference string to be acquired. During simulation the author compared MPDRC with FIFO, LRU and LFU replacement algorithms with the proposed innovative page replacement algorithm (MPDRC). It was discovered that MPDRC algorithm has 8.34% better than FIFO, 16.63% better than LFU and 10.6% better than LRU algorithm. Overall average this work innovative page replacement examined 11.85% better than existing algorithm. After comprehensive simulation experiments is summarized that for proxy caching the MPDRC hit ratio performance better than others existing algorithms.A study by Paajanen (2000), worked on Least Recently Used (LRU). The algorithm takes into considered the imbalance of the cost of read and writesmemory when replacing pages. The algorithm defines that, the string been used in the list was replaced in the needed frame. The LRU algorithm has better performance whencompared with FIFO.

III. Method

In this study, three standard algorithms were considered; FIFO, LRU and OPT

i. **First in First out (FIFO):** It replace the page at the head of the queue when a page is brought into memory. FIFO associates with each page, the time the page was brought to the tail of the queue (Abraham, et al., 2012).
ii. **Optimal:** In operating system the algorithm swaps out the page whose next reference will take longer time for future reference. For example, a page that is not going to be used for the next 6 seconds will be swapped out over a page that is going to be used within the next 0.4 seconds(Khulbe, et al., 2014).
iii. **Least Recently Use (LRU):** This based on the probability theories in a logic idea. This idea consists of pages which have been heavily used in the last few instructions, which will probably be used again in the next few pages that are not use for a long time may say in this state again.

The following are some of the lapses observed with the existing algorithms

i. The optimal algorithm fetches the string that will be used in the future, whereas the operating system cannot predict the future occurrence. Therefore, the optimal is not a good choice
ii. FIFO algorithms fetch and replace the string that came into the frame first. This operation is based on first in first out (queue). As a result of this operation, more page fault is generated. Hence, FIFO is not a good choice.
iii. LRU, this algorithm is similar to FIFO just that it fetches in the page that has been used the least, this is better than FIFO but still have high page fault than OPTIMAL.

Hybrid Page Replacement Modeling

The HRA comprises of two standard page replacement algorithms namely, LRU and Optimal. These where hybridized to give a better performance in terms of number of page faults generated when compared to the individual paging algorithm. As shown in fig.1, the proposed HRA algorithm is designed for the purpose of less page fault in the page replacement algorithm.Knowledge of LRU and OPT algorithm is crucial in the usage of HRA algorithm. The victim page is been swapped in/out of the physical to the backing store (virtual memory) using HRA technique. The frame is shared between OPT and LRU i.e. the first frame is given to OPT, and it takes the first reference string, processed it with the frame and swap out the particular frame that will not be used for the longest period of time and LRU will pick the second reference string, process it with the frames and swap out the particular frame that has been used least recently among all the frames on the pages. Java SE was used to implement the algorithm and the units that were tested includes: the first dialog box for the input frame, the second for the length of the strings and the third for the reference strings itself.

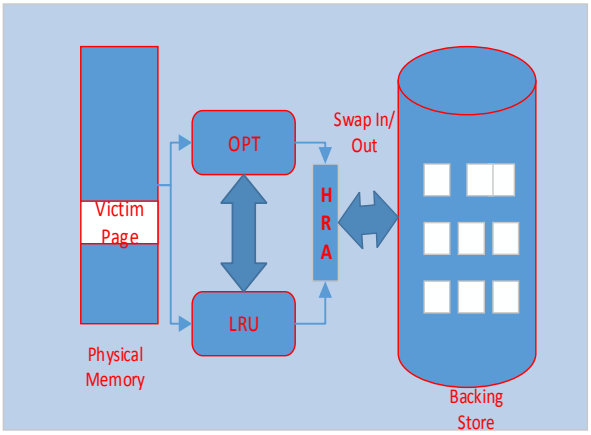

Fig. 1: Proposed HRA Model

IV. Results and Discussion

In this study, the results achieved are the outcome of simulation of the designed system. This result is the evaluation performance metric of the proposed HRA system in terms of NUNBER OF PAGE FAULTS generated. Fig. 2 depicts the simulation interfacewhere the experiments was conducted. Three (3) cases

with reference strings(70120304230321201701, 470710121271 and 12341251234) were considered with3,4,5,6 and 7 frames on each.

Case 1: In this case, reference string 70120304230321201701 was simulated on 3, 4, 5, 6 and 7 frames. The result in fig. 3 depicts the performance of HRA when compared with FIFO, LRU and OPT. It shows that HRA outperformed among others in terms of lower number of page fault generated as the frame number increases from 4 and above.

```
Total number of page fault : 6
Do you want to continue? if Yes, press 1

Menu
1.FIFO
2.LRU
3.Optimal
4.hybrid
5.Exit
Enter your choice : 4

String is : 1
String is : 2
String is : 3
String is : 4
String is : 1
String is : 2
String is : 5
String is : 1
String is : 2
String is : 3
String is : 4
String is : 5
Total number of page fault : 5
Do you want to continue? if Yes, press 1
```

Fig. 2: simulation console

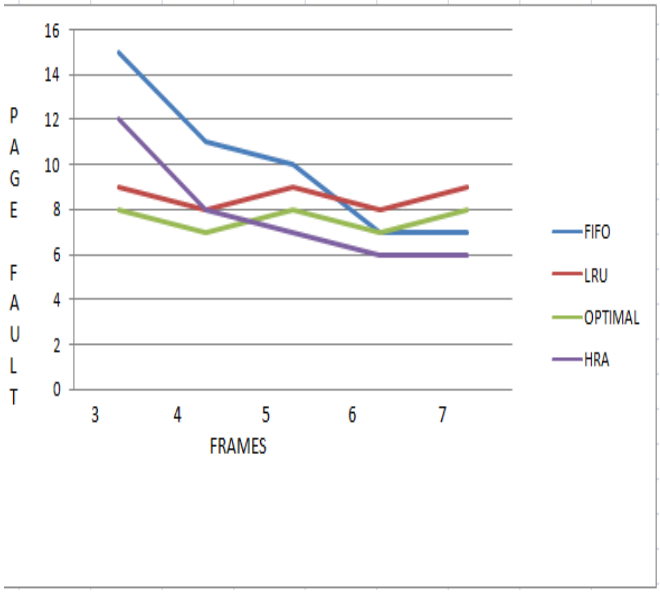

Fig. 3: Graph of HRA with 20 reference strings.

Case 2: In this case, reference string 470710121271 was simulated on 3, 4, 5, 6 and 7 frames. The results in fig. 4 depicts the performance of HRA when compared with FIFO, LRU and Optimal. It was observed that, with few reference strings, HRA have similar performance with OPT but still outperformed OPT from 4 frames and above.

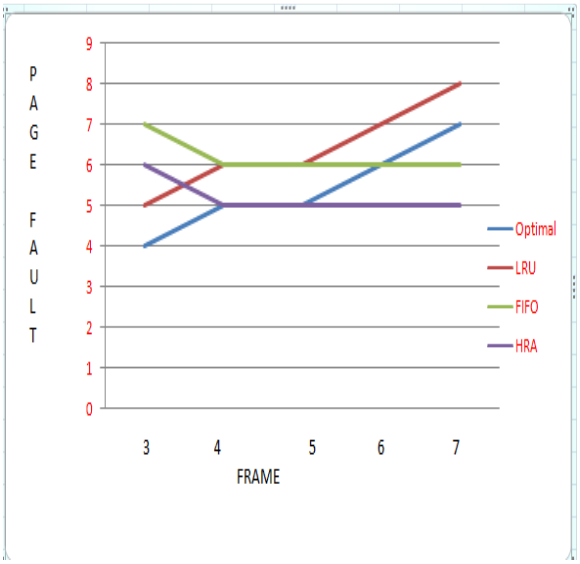

Fig. 4: HRAwith 12 reference strings.

Case 3: In this case, reference string 123412512345 was simulated on 3, 4, 5, 6 and 7 frames. The results in fig. 8 shows thatHRA have poor performance at the lower number of frame, but from 4 frames and above, it outperformed others in terms of lower number of page fault generated.

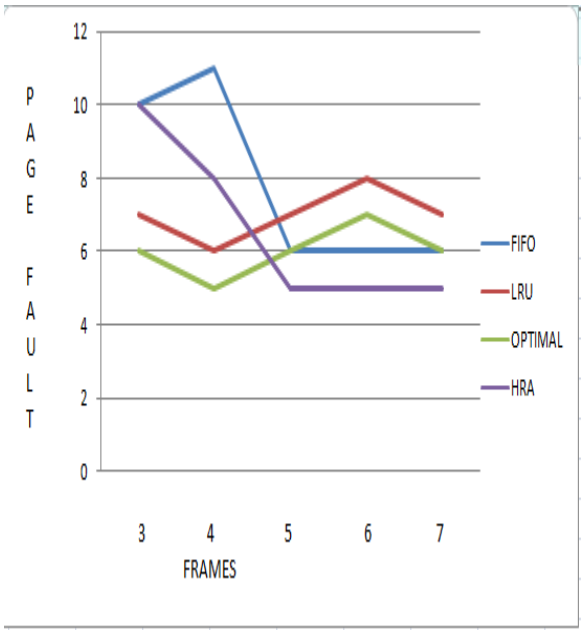

Fig. 5: HRA with 12 reference strings.

V. CONCLUSION AND FUTURE WORKS

An effective page replacement algorithm called HRA which utilized the concepts of LRU and OPTIMAL algorithms wasdeveloped and tested. Through the different tests, the results showed that HRA simulation is superior to all existing page replacement intended for Operating System as per page fault.Extensive research can be done to improve the HRA on lower number of frames.

REFERENCES

[1] Abraham,S. Peter,B.G. and Gred, G.(2012). Operating System Concepts.page 397-463.2

[2] Andrew, S.T. (2015). Operating systems: design andimplementation. *New Jersey: Prentice-Hall, Inc*.5(3):519-540

[3] Anthony, S. and Zhang,X. D. (2015).LIRS: an efficient low inter-referencerecency set replacement policy to improve buffer cache performance Perform Eval., 1(30): 31-42.

[1]

[4] Brinkhoff, H.(2002).CLRU: A New Page Replacement Algorithm for NAND Flash-based Consumer Electronics, 43.

[5] Desseau,C. C and Parmar, S. S. (2008). A comparison of page replacement algorithm.*IACSIT International Journal of Engineering and Technology.*3(2):32-36

[6] Duesseau, A. (2008) .Paging. 75-79. retrieve from https://careerride.com/os-paging.aspx

[7] Genta, R. Erand, E.andIgli,T. (2015). A Comparison of Three Page Replacement Algorithm: *FIFO, LRU and OPTIMAL.* retrieve from https://www.mscer.ogr/journal/index.php/ajis/article/download/7405/7089

[8] Hwang, H. and Jotwani W. (2011) Energy-Aware Flash memory management in virtual memory system. *IEEE Trans, VLSI Syst.*16(8): 403-412

[9] Jananee,S. (2014). Page replacement, International Journal of Computer Science and Information Technology Research ISSN 2348-120X 2(3): 90-99 . retrieve from www.researchpublish.com.

[10] Jung, W. C. and Parmar, S. S. (2013).Performance analysis of LRU page replacement algorithm.*International Journal of Engineering Research and Applications* (IJERA) 3(1): 2070-2076.

[11] Kavaraid D.(2013). LRU-MRU with physical address cache replacement algorithm on FPGA

application.in*Computational Science and Engineering (CSE), 2014 IEEE 17th International Conference on*.1302-1307.

[12] Khulbe,S.Bancha,N. and GITA,B. (2014). Analysis andPredictability of Page Replacement Techniques towards OptimizedPerformance, IRCTITCS. 16, 13-15.

[13] O'Neil, E.J. O'Neil, P.E, and Weikum, G.(1993).The LRU-K pagereplacementalgorithm for database disk buffering.*SIGMOD.* 2(22):297-306

[14] Paajanen H. (2007). Page replacement in operating system memory management.*Master's Thesis in Information Technology*.University of Jyväskylä

[15] Park,S., Jung, D., Kang, J. (2006). Cflru: a replacement algorithm for flash memory. *Proceedings ofthe 2006 international conference on Compilers*.

[16] Priyadharshni,A.Nayak, K. R, Vora,K. D,Purohit,M. D and Chawan,P. M,(2016).Introduction to algorithms.third edition. *MIT press*. 2(6) :81-90

[17] Ronald,J. (2006).Page replacement in operating system memory management.*Master's Thesis in Information Technology*. University of Tirana, Department of Computer Engineering.

[18] Shen,X. Hyun, T. Kim, and Bahn H.(2004). A compressed file systemmanager for flash memory based consumer electronics devices, IEEETrans. *Consum. Electro*.59(3): 544-549.

[19] Sumit, C. Srisa-An, W. Chang, J.M. (2015).A study of page replacement performance in garbage collection heap. *Journal of Systems and Software*. 3(58): 235 – 245.

[20] Tanenbaum A. S. (2008). Modern Operating Systems. *Pearson International Edition*.2(4): 235-245.

[21] Withington, Q. Hu, W. C. Lee, and D. L. Lee.(2001) Performance Evaluation of an Optimal Cache Replacement Policy for Wireless Data Dissemination,*IEEE Transactions on Knowledge and Data Engineering,* 16(4): 125-139.

