COMPARATIVE ANALYSIS OF REPLACEMENT ALGORITHMS TECHNIQUES REGARDING TO TECHNICAL ASPECTS

Muhammad Waqar
COMSATS Institute Of information technology Islamabad
PAKISTAN
mwaqar@comsats.edu.pk

Anas Bilal
University Of Lahore
Islamabad Campus
PAKISTAN
chanasbilal@gmail.com

Ambreen Malik
University Of Lahore
Sargodha Campus
PAKISTAN
Amber920@gmail.com

Imran Anwar
University Of Lahore
Sargodha Campus
PAKISTAN
Info.vstudents@gmail.com

ABSTRACT

Computer Operating System are used page references for the Memory management ,there are different types of page replacement algorithms that decides which pages are removed from the memory when more reference pages are allocated in the memory. When a free page is unable to use allocated memory then page fault is occurs this occurrence of page fault is known as paging. Paging occurs due to less space or available space is less than the required memory space. In this paper we encapsulate the major types of the Page replacement algorithems that are proposed in the latest researches. We elaborate the commonly used algorithems such as FIFO, Beladys MIN, The Optimal Algorithm, LRU, LRU Approximation, Hybrid LRU, CLOCK Algorithm, Dueling CLOCK,LRIS, CLOCK-Pro, and ARC. This paper also we also compare the LRU with the hybrid LRU and find the comparative results in this comparison and also compare LRU with the FIFO,LRU and optimal algorithms to with the help of examples of these algorithms find that which one is better in performance ,our main focus is to analyze the LRU,FIFO and optimal algorithms results when number of frames are increased in and due to increment of frames its also effected on page fault and we want to determine the HIT-ratio in all of these cases.

Keywords: FIFO, Beladys MIN, The Optimal Algorithm,LRU, LRU Approximation, Hybrid LRU, CLOCK Algorithm, Dueling CLOCK, LRIS, CLOCK- Pro, ARC, PSEL, IRR, LIR, HIR.

INTRODUCTION

In OS main issue is to cater the resource management using the system software's [1].OS also determinate the processing arrangement's and classified the operation in a proper way. Work without the commands of users and its perform the operation till the end[2].the main task of the operating systems is manage the memory and specify the operations and interact with the prevent process that is placed in the OS. Memory managements is the main part of OS in which all the operation that is related to memory is performed. Now a days multi programs loaded in same time and these programs don't interaction with other programs due to agent systems are more reliable .To perform multi-tasking there are different techniques such as paging-technique is one of them in which physical memory is used and dived this memory in small blocks with same size called frames and it's also called as pages[3][4].During the processing of an operation the pages transport to the main memory and the main memory keep several operations at a same time with the help of virtual memory techniques that are start of the cycle the results must be save in the memory used by the OS. In this process the required pages are transferred using page replacement algorithms. These pages replace with the paging replacement algorithm's such as FIFO,LIFO,LRU,NRU,CLOCK,AGING,OPTIMAL,SECOND-CHANCE,DUELING-CLOCK, LIRS,CLOCK PRO,LRU-K Algorithm's etc.[5].During the processing in page replacement algorithm's efficiency of the operations is the main issue to overcome this issue assigns the memory to the pages to increase efficiency of the systems. During the operation to avoid the pages disorder it's compulsory to follow the page replacement procedure. In [4] author's proposed a new Page replacement algorithm technique that's is known as Token ordered LRU to decrease the errors in these algorithm's .To make this techniques more

efficient they use LINUX kernel that is more reliable to indicate errors and replace these with local page frame.

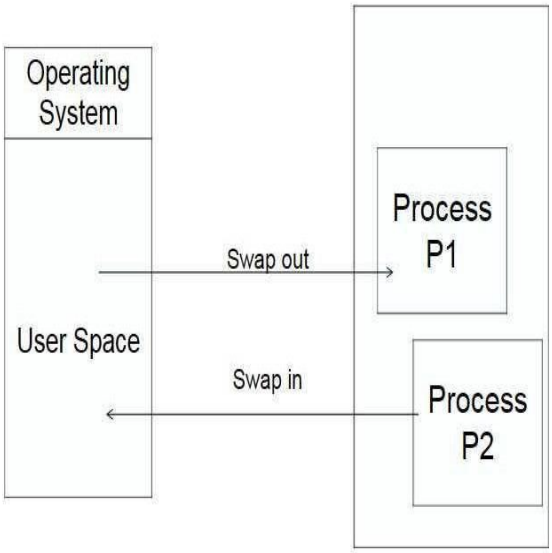


Fig. 1. Swapping in Algorithm.

Later on in [1] authors describe the Odd-Back-Even-Forward (OBEF) for improvement in page algorithm's that replace the page in the-memory.This algorithm is working on even and odd cycles, at the After that the previously stored results are removed from the memory in second cycle. It's also updated form of FIFO,LRU,LFU,Aging,Clock & random algorithms and is more efficient then these algorithms. The algorithm base on Fast Fourier Transform (FFT) is proposed in [6] to replace the previously used pages in the memory. Its increase the data rate of the algorithm using the cache memory of the system. Due to usage of cache memory the processing speed is going to be very fast and efficient and save more time than previous techniques. Authors also compare their proposed techniques with the previously proposed techniques and they conclude that their proposed technique has less error than the others and its time saving and more efficient than the other techniques. In [7] authors discussed about the efficiency of the LRU algorithms in their research they find that the efficiency of LRU is improved with the help of RAM and cache memory that are used simultaneously and maximize the number of pages and errors.

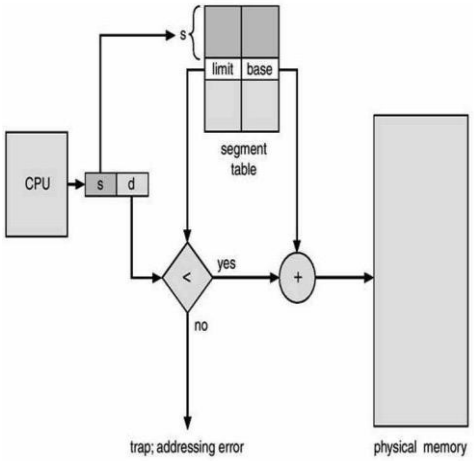


Fig. 2.Segmentation in Algorithm.

The pages that are stored in the RAM and not used in the forthcoming time these pages are removed from the RAM and stored in the cache memory for further use. It's also noticed that if the cache size is large

then it's affected on the data rate of pages and decrease the error rate. In [8] researcher proposed partition based replacement algorithm that help to increase the page replacement with the help of cache memory .Its working is similar to FIFO Algorithm but this technique divide the cache memory in two parts A1 and A2 .In PDB number of pages are also divided in two parts to increase the processing data rate .This algorithms is more efficient than the previously proposed techniques. Authors used FL (Fuzzy Logic) that replaces the pages from the page replacement from the main memory.

FL is used to make a decision that which pages are transferred to the memory on the basis of these decision authors concludes the results that this technique is better than the LFU and LRU algorithm techniques [9].

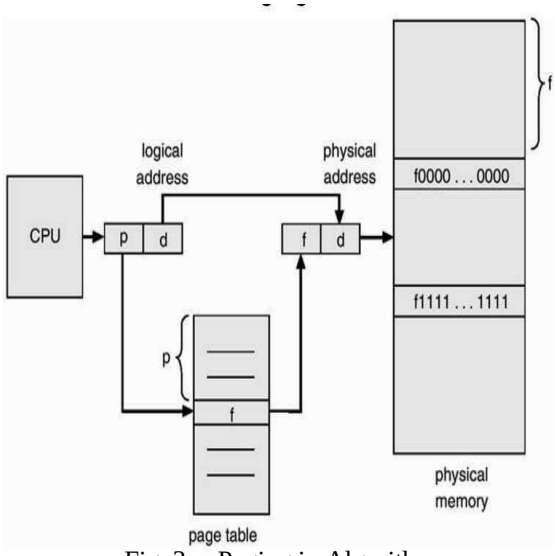


Fig. 3. Paging in Algorithm

II.PAGE REPLACEMENT ALGORITHMS

Pare replacement algorithms is used by OS to make de-cisions about the memory selection and make decision that which pages are removed or which pages are added in the memory. When an error accurse and the free pages are not used for memory allocation that is known as paging & page fault. Replacement algorithm is working on the information that is provided by the hardware of the OS .There are many types of replacement-algorithms some of them are described below.

A. First in, first out (FIFO) Algorithm

First-In-First-Out(FIFO) algorithm is the most simplest types of replacement algorithm and also very oldest-algorithm. Working phenomena of FIFO is to replace a new page with the earliest page from the all pages that are in the main memory.FIFO target the pages that are in the memory from a long time rather than the pages that are used recently. Pages are ordered in Queue with respect to their time off arrival. InFIFO page in used only one time and stored in the memory for long time that's why this is not a good algorithm if we increase the memory of the algorithm the error rate will be increase badly[5].

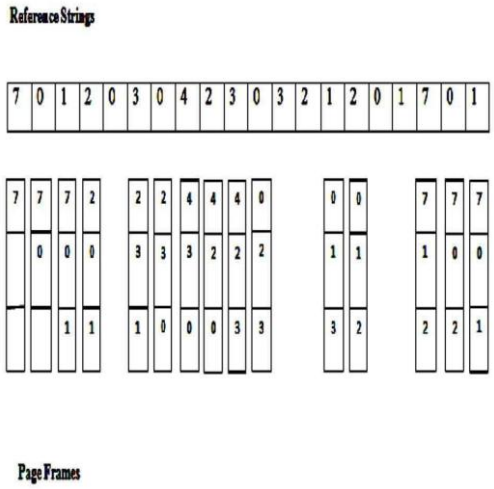


Fig. 4.FIFO representation.

B. Beladys MIN Algorithm

Belady's MIN is the best algorithm with the excellent performance. Its remove unnecessary page from the memory that are not used for the long time. Replacement of pages are based on the coming sequence of pages but **it's** not possible to predict the sequence of coming pages sequence itself **that's** shy this algorithm makes systems speculative for the real time systems. This algorithm is used to compare the algorithms and find which one is more effective.

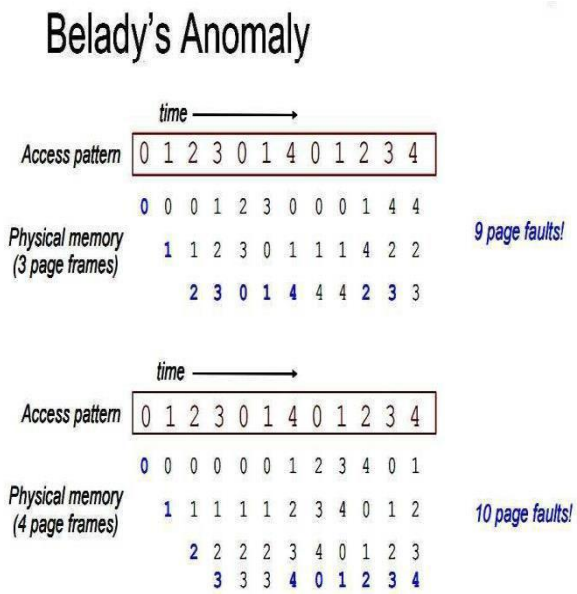


Fig. 5.Beladys Anomaly.

C. The Optimal Algorithm

Optimal Page replacement Algorithm has good performance.This algorithm have very less Error rate that why it is the best algorithm among the other algorithms but it is not implemented in real time systems till-now. In this algorithm the page that is never used again is replaced with the new page. The pages that are in the queue are following the queue till end. The page that is not used for a long time the algorithm replaces this page to clear a memory. The issue face during implementation of this algorithm is that it's required future predictions of the strings[10].Example of Optimal algorithm is shown below. The page is

sleeted in example that will not require in the future from the long time .Page errors are shown in the last step of the example the optimal algorithm is more better than the other due to less error rate.

Time		0	1	2	3	4	5	6	7	8	9	10
Requests			c	a	d	b	e	b	a	b	c	d
Page Frames	0	a	a	a	a	a						
	1	b	b	b	b	b						
	2	c	c	c	c	c						
	3	d	d	d	d	d						
Page faults							X					

Fig. 6. The Optimal Algorithm(First Step).

Time		0	1	2	3	4	5	6	7	8	9	10
Requests			c	a	d	b	e	b	a	b	c	d
Page Frames	0	a	a	a	a	a	a	a	a	a	a	
	1	b	b	b	b	b	b	b	b	b	b	
	2	c	c	c	c	c	c	c	c	c	c	
	3	d	d	d	d	d	e	e	e	e	e	
Page faults							X					X

Fig.7. The Optimal Algorithm(Second Step).

Time		0	1	2	3	4	5	6	7	8	9	10
Requests			c	a	d	b	e	b	a	b	c	d
Page Frames	0	a	a	a	a	a	a	a	a	a	a	a
	1	b	b	b	b	b	b	b	b	b	b	b
	2	c	c	c	c	c	c	c	c	c	c	c
	3	d	d	d	d	d	e	e	e	e	e	d
Page faults							X					X

Fig. 8.The Optimal Algorithm(Third Step)

D. Least-Recently-Used(LRU)

In this algorithm type replace the page with the page that is not used for the log time and take place in the main memory. This algorithm is more efficient than the FIFO because store a pattern of programs performance and the pages that are used in the pas are again sometime for short time .It also contain the previous record of the pages that are used before. Due to huge overhead on the system implementation of

LRU is on a risk[11].Example is shown in fig below in which LRU keep the record when system used a page and replace them with the recently used page.

Time	0	1	2	3	4	5	6	7	8	9	10
Requests		c	a	d	b	e	b	a	b	c	d
Page Frames 0	a										
1	b										
2	c										
3	d										

Fig. 9. Least-Recently-Used(1st step).

Time	0	1	2	3	4	5	6	7	8	9	10
Requests		c	a	d	b	e	b	a	b	c	d
Page Frames 0	a	a	a	a	a						
1	b	b	b	b	b						
2	c	c	c	c	c						
3	d	d	d	d	d						
Page faults						X					

Fig. 10. Least-Recently-Used(2nd step).

Time	0	1	2	3	4	5	6	7	8	9	10
Requests		c	a	d	b	e	b	a	b	c	d
Page Frames 0	a	a	a	a	a	a	a	a	a		
1	b	b	b	b	b	b	b	b	b		
2	c	c	c	c	c	e	e	e	e		
3	d	d	d	d	d	d	d	d	d		
Page faults						X				X	

Fig. 11. Least-Recently-Used(3rd step).

Time	0	1	2	3	4	5	6	7	8	9	10
Requests		c	a	d	b	e	b	a	b	c	d
Page Frames 0	a	a	a	a	a	a	a	a	a	a	
1	b	b	b	b	b	b	b	b	b	b	
2	c	c	c	c	c	e	e	e	e	e	
3	d	d	d	d	d	d	d	d	d	c	
Page faults						X				X	X

Fig. 12. Least-Recently-Used(4th step).

Time		0	1	2	3	4	5	6	7	8	9	10
Requests			c	a	d	b	e	b	a	b	c	d
Page Frames	0	a	a	a	a	a	a	a	a	a	a	a
	1	b	b	b	b	b	b	b	b	b	b	b
	2	c	c	c	c	e	e	e	e	e	d	
	3	d	d	d	d	d	d	d	d	c	c	
Page faults							X				X	X

Fig. 13. Least-Recently-Used(5th step).

E.LRU Approximation Algorithm

In approximation LRU page is represented as a bit and the initial value of a bit is 0 and the reference page bit is represented as 1.In this algorithm the order of the bits is unknown, if ones are present in the queue then these ones are replaced with the 0.Furthermore more bits are added in the queue as a reference bits and these bit shift the bits in right side by 1 and replace the bits with the high order bits 1 if **it's** called other vise it replace with 0.Lowest-refrence bits with the value 1 are the east-recently used bits and these are replaced with the other bits. Example of Approximate LRU is shown below.

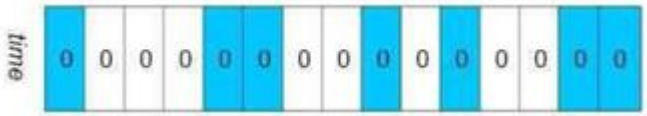


Fig. 14.Approx LRU(1st step).

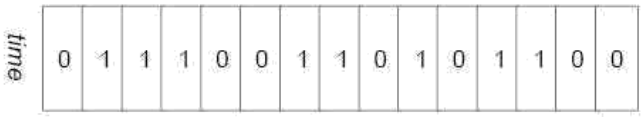


Fig. 15.Approx LRU(2nd step).

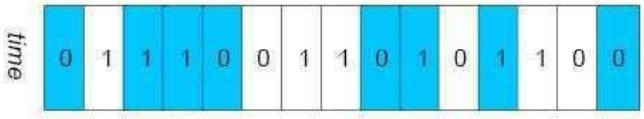


Fig. 16.Approx LRU(3rd step).

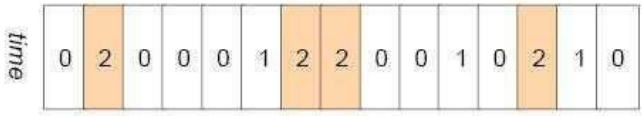


Fig. 17.Approx LRU(4th step).

F. Hybrid LRU Algorithm

Hybrid LRU algorithm is totally based in LRU algorithm. In Hybrid LRU more feature are introduced such as number of all references among the all pages is calculated on their referred pages. Modified Reference is also used in this algorithm .I.e when a page is modified then assign a value to its reference.Hybrid-LRU representation for 3-frames are shown in fig below.

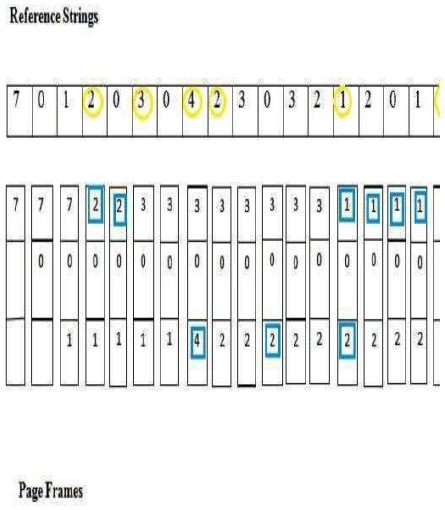


Fig. 18. Hybrid LRU.

G. Not Frequently Used(NFU) Algorithm

In NFU algorithm the hardware of the OS can't support the LRU and the all reference pages are set to 0 initially. When a reference page is added the counter added the reference bit into the page bit will be 0 or 1 Shift to right MSB=R.

H. CLOCK Algorithm

In clock algorithm we imagine that all the framer of pages are presented as a circular-list that similar to a Clock. Oldest page in the buffer is represented by a hand of clock .Each page is represented as a bit and bits are assigned to the pages during the referencing procedure is processing on the page. The page is replaced when system found the missing page in the list in this case earliest page is used by the hand. In this situation the bit of the reference page become zero and then its moved to the next oldest-page and its working in a queue till find a bit with the value 0.Removed the page with reference bit 0 and replace with the other reference page, Clock algorithm is used to approximate LRU in very efficient way to decrease the fault rate.

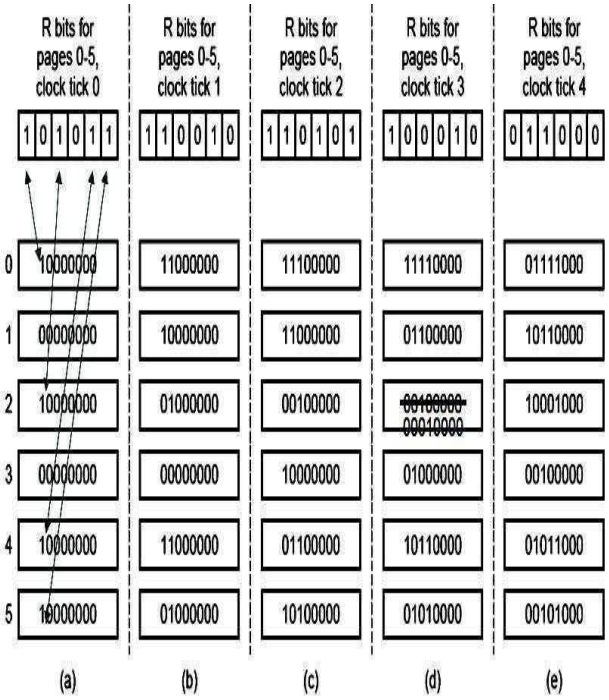


TABLE I
NFU.

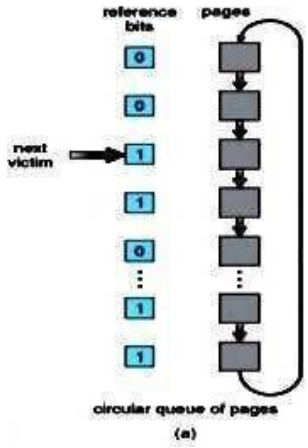


Fig. 19. CLOCK Algorithm.

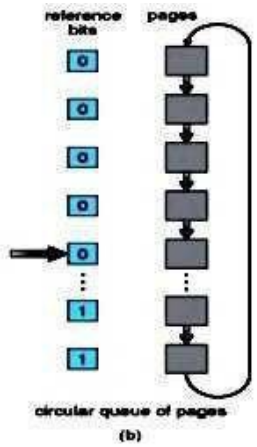


Fig. 20. CLOCK Algorithm.

I. Dueling CLOCK

Dueling clock algorithm has adaptive properties and it's totally based on clock algorithm. The main drawback of the clock algorithm is not a scan-resistant [12].To overcome this issue Clock-algorithm presented a dueling clock approach that use adaptive properties of the clock for the page replacement. Furthermore to make clock more efficient change required in the clock in hand-of, hand should always point to the new page from the buffer rather than the old one. In dueling clock algorithm cache s divided into three-groups G1,G2&G3.G1 is the smallest group that is used for the page replacement. 2nd group G2 is used to make a clock scan resistant and the 3rd group G3 is the largest group and this group also scan resistant and its depend on the performance of the other two groups .PSEL is used to evaluate the page replacement for G3 group.

J. Low Inter-reference Recency Set (LIRS)

The LIRS are recently in the Inter-Reference pages for eviction in the dominant factors[13].The IRR is used to refer a page that are access continuously between the two-references of that page. If the size of the current IRR is large then it is suitable for the next ejection as per Belady's-MIN algorithm.The page selected for the IRR is recently used algorithm and its make a difference between the HIR and LIR. Cache kept the large number of LIR pages and some of the HIR pages are also stored in the cache .If cache if full then the HIR pages are removed from the cache to make a space for LIR pages.if we have 100% storage of cache the memory used by the HIR pages are only 1% of the total memory and rest of the cache memory is used by the LIR pages.An example of LIRS is shown below.

$L_{lirs}=2$, $L_{hirs}=1$

V time / Blocks	1	2	3	4	5	6	7	8	9	10	R	IRR
A	X					X		X			1	1
B			X		X						3	1
C				X							4	inf
D		X					X				2	3
E									X		0	inf

LIR block set = {A, B}, HIR block set = {C, D, E}

Fig. 21. LIRS Example(1st step Mapping to Cache)

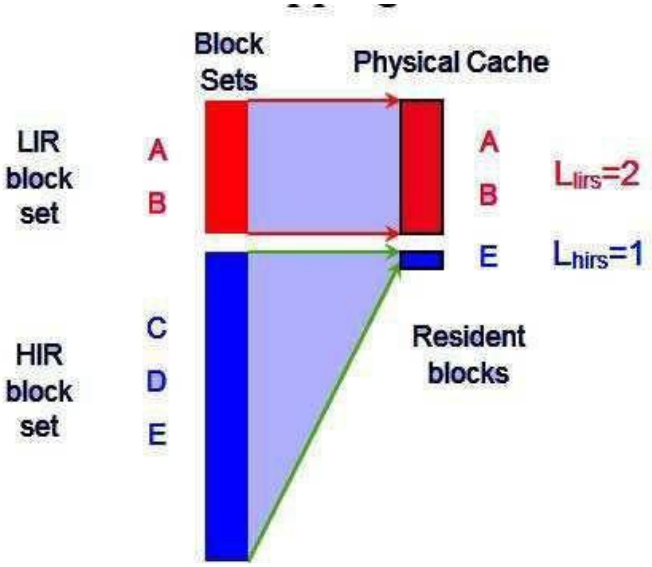


Fig. 22. LIRS Example(2nd step Replace a HIR Block).

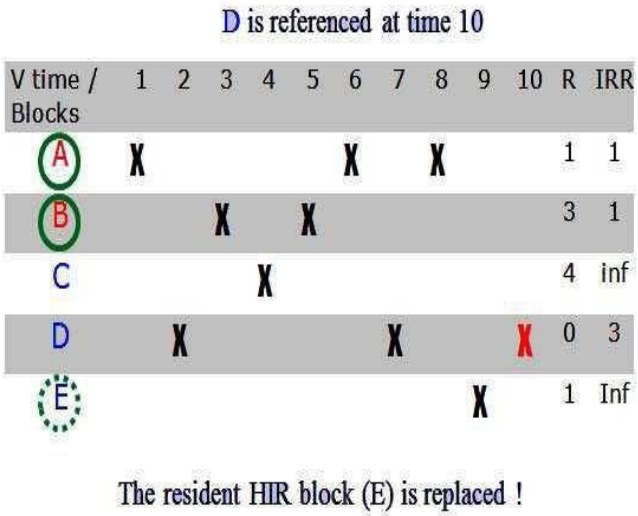


Fig. 23. LIRS Example(3rd step Recency of LIR Block Used).

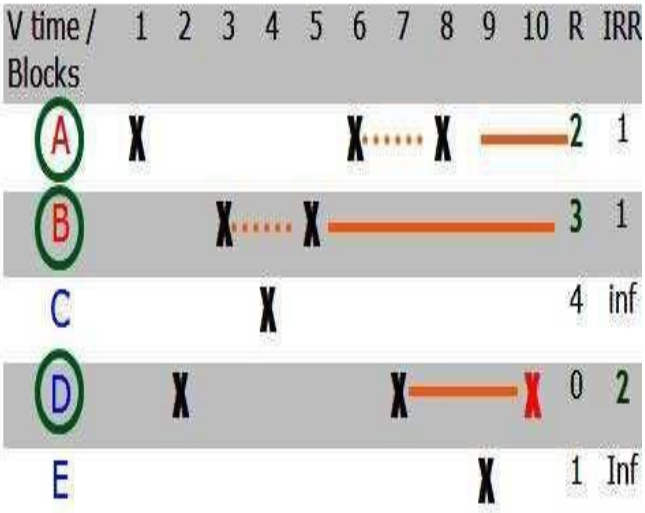


Fig. 24.LIRS Example(4th step).

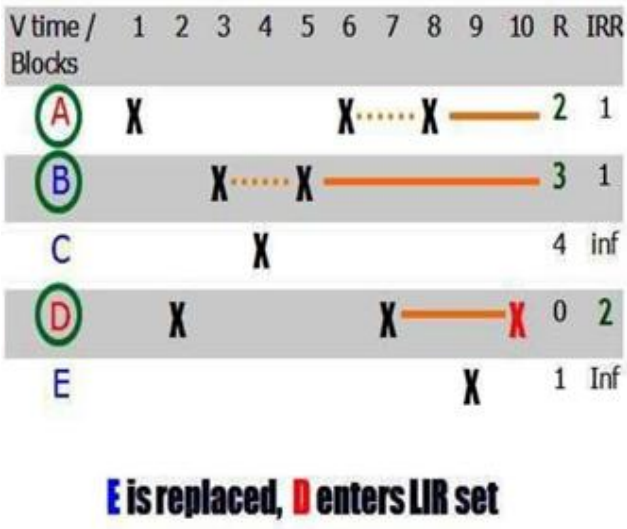


Fig. 25.LIRS Example(5th step).

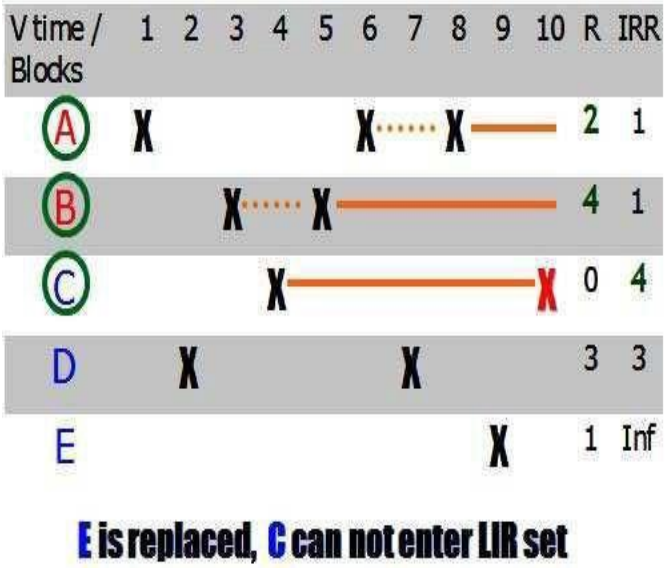


Fig. 26.LIRS Example(6th step).

K.CLOCK-Pro

The clock pro is used to approximate the results of the LRIS algorithm with the help of clock. Clock pro reuse the distance which is used to compare the IRR with respect to LIRS, which is the most important part for the page replacement in this algorithm [14]. When the reference page is accessed in the Clock pro algorithm reuse-distance is used for the limited time period and accesses the pages that are not accessed from t long time .pages are categorized in two categories one is cold and second one is hot page. Cold page has a large number of reuse distances and hot page has less reuse distance as compare to the cold Page.

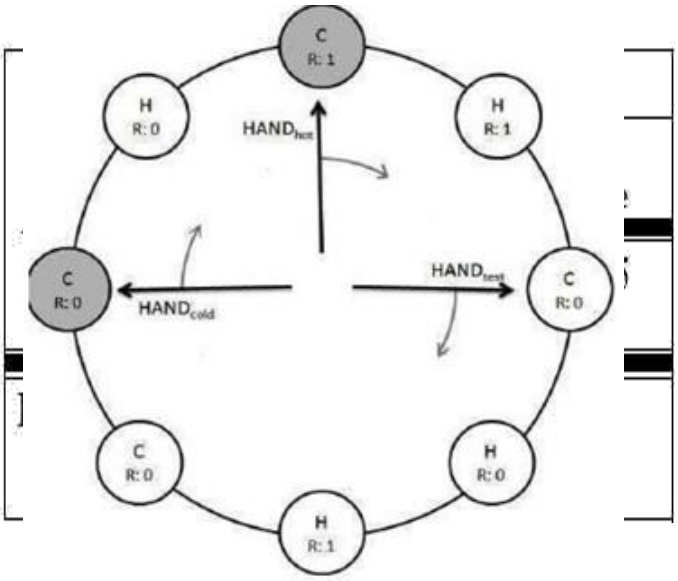


Fig. 27.Working of Clock Pro.

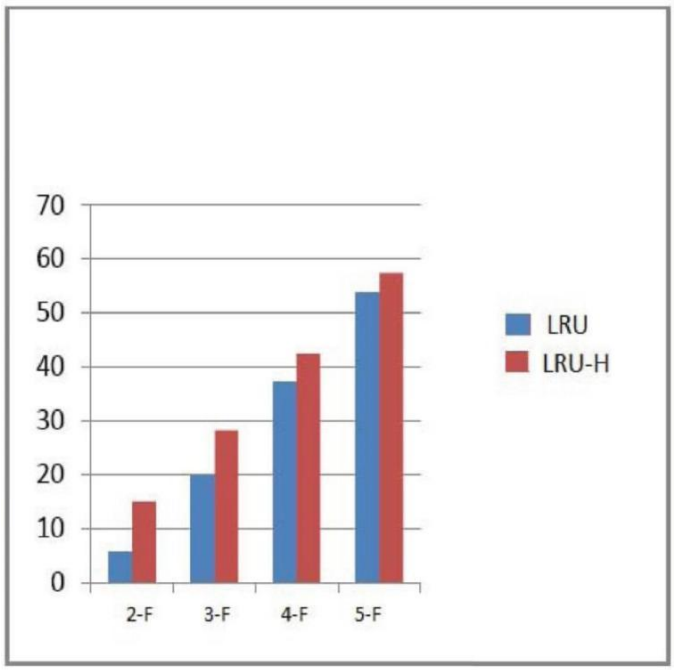
L. Adaptive Replacement Cache (ARC)

The Adaptive-Replacement-Cache flexible page- replacement algorithm that is developed in IBM labs in almaden.In this algorithm both types of pages that are used frequently & recently are keep tracking by the ARC also save the history of these ages[15].Two list of LRU is maintained by the ARC, in list 1 all the record of the recently accessed pages that are viewed once and in the second list maintain a data of the pages data that are accessed twice. List one is used for the short-term-recensy and the second list is used for the long term-frequency. Cache memory is also divided into two categories to and bottom cache ghost entries. The list one divided into T1&B1 and second list is divided into T2&B2 respectively.

III.COMPARISON OF LRU & HYBRID LRU ALGORITHMS

In this Section we use simulation tools to find the page fault and fault Ratio on the basis of these results we con-clude that which algorithm is more reliable than the other algorithm. For this experiment we generate random sequences for LRU & hybrid-LRU ,length of the randomly gener-ated sequence is 20.In this experiment we generate four-cases for the memory allocation these are frame-2,frame-3,frame-4,frame-5 and these results are shown in table 1 Bar diagram is also shown belowas a BAR 1 on the basis of these results. Formula that is used to find the HIT-ratio is described as. HIT-Ratio=Number-OF-Request-In-Cache/Total-Number-OF-Requests to find a percentage we use Percentage-HIT=Ratio100 the results show that the LRU-H is more reliable than the LRU algorithms, L R U -H have the more efficient results than the LTU algorithm.

TABLE II
COMPARISON OF LRU&LRU-H.



BAR I
WORKING OF CLOCK PRO.

IV. COMPARISON OF FIFO,OPT&LRU ALGORITHMS

In this section we compare the FIFO, OPT&LRU with respect to their page-fault-rate and Hit-ratio by taking a random string for page referencing. Let the page reference string is 7 , 0 , 1 , 2 , 0 , 3 , 0 , 4 , 2 , 3 , 0 , 3 , 2 , 1 , 2 , 0 , 1 , 7 , 0 , 1 Now we can solve each of the Algorithm with this string and conclude the result for each algorithm.

A. FIFO

In FIFO page are arranged in the queue with respect to their time of arrival .FIFO remove the page which is stored in the memory for the long time. Now we solve the example for the FIFO using the proposed referencing string and a page fault are indicated by a tick.

FRAME-1: There are 20 faults in single memory page in this algorithm same reference-page will never referred-twice in a single row, fault is produced by a single page reference.

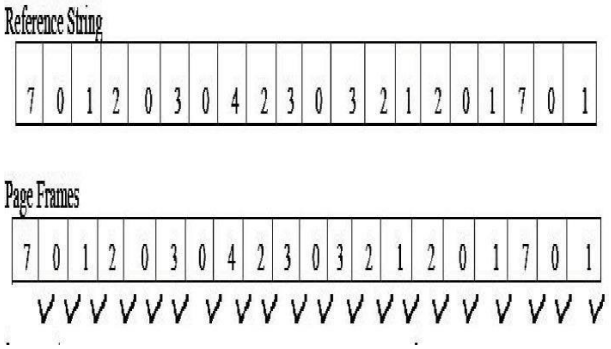


TABLE III
FRAME-1 REPRESENTATION.

Frame-2: 15 faults in a two page-memory

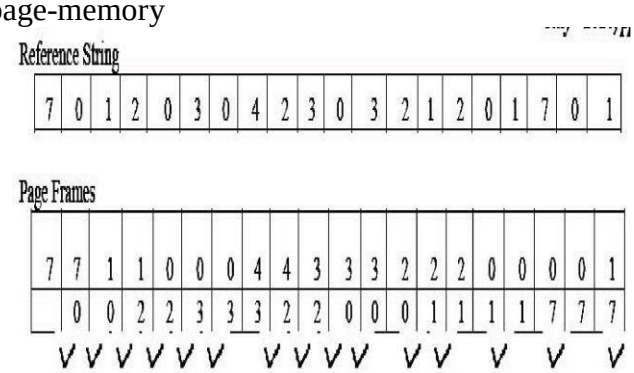


TABLE IV
FRAME-2 REPRESENTATION.

Frame-3: 15 faults in a three page-memory.

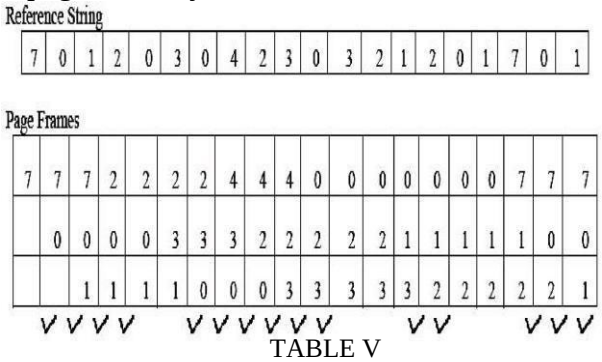


TABLE V
FRAME-3 REPRESENTATION

Frame-4: Ratio of faults decreased to 10 in a four page-memory.

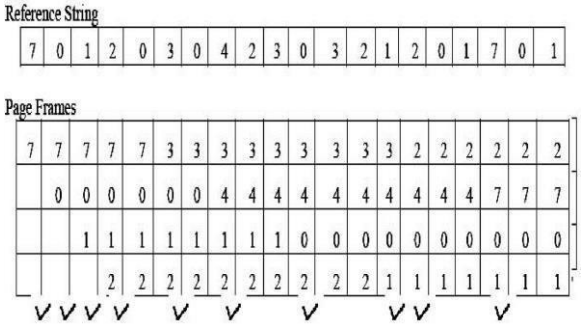


TABLE VI
FRAME-4 REPRESENTATION.

Frame-5: In last frame-5 there are only 09 faults in the five page-memory

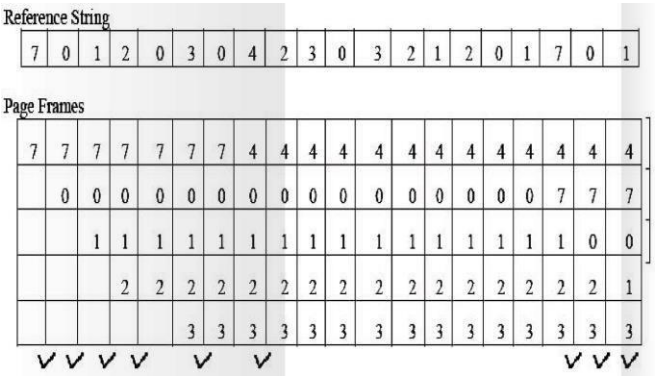


TABLE VII
FRAME-5 REPRESENTATION

B. OPTIMAL

In this algorithm techniques page is replaced with the unused page that is not required in the future,

Frame-1: 20 faults in a single memory-page

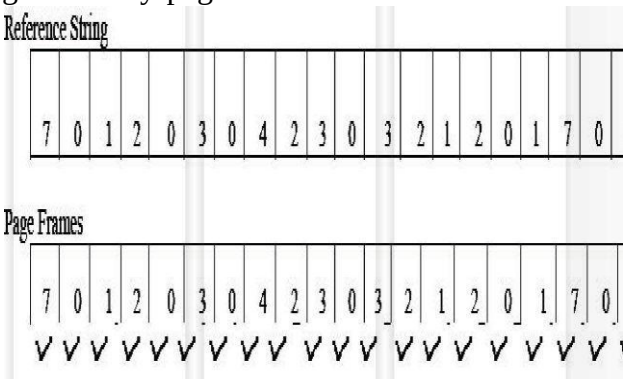


TABLE VIII

FRAME-1 REPRESENTATION OF OPT

Frame-2: 13 faults in a two page-memory.

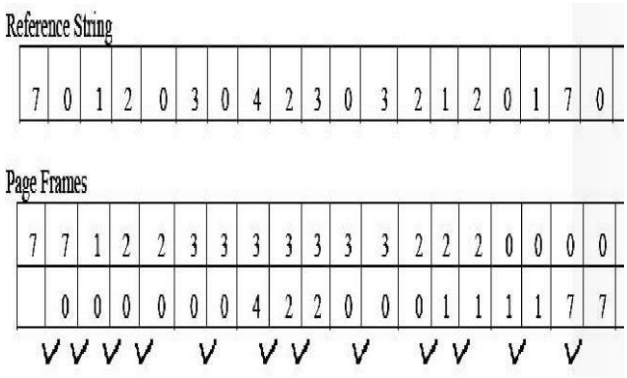


TABLE IX
FRAME-2 REPRESENTATION OF OPT

Frame-3: 09 faults in a three page-memory.

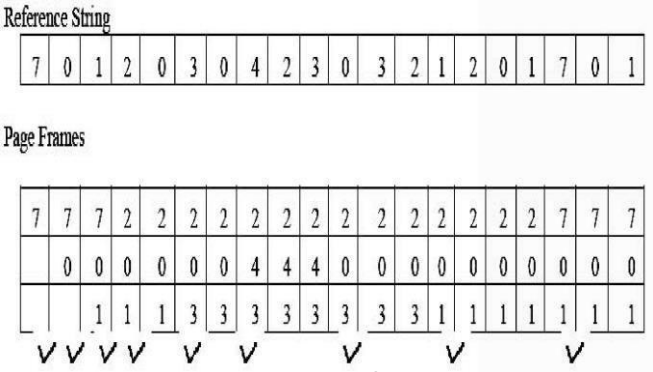


TABLE X
FRAME-3 REPRESENTATION OF OPT

Frame-4: 08 faults in a four page-memory.

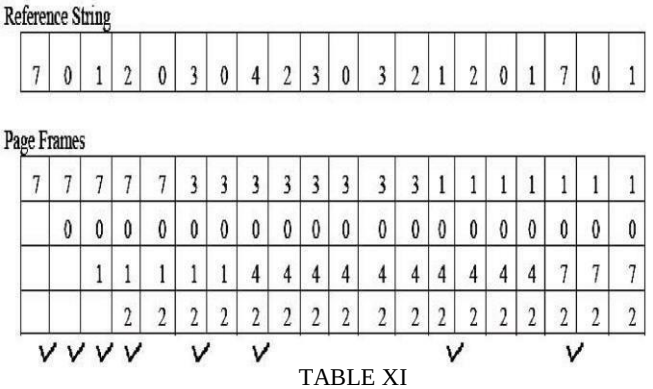


TABLE XI
FRAME-4 REPRESENTATION OF OPT

Frame-5: 07 faults in a five page-memory.

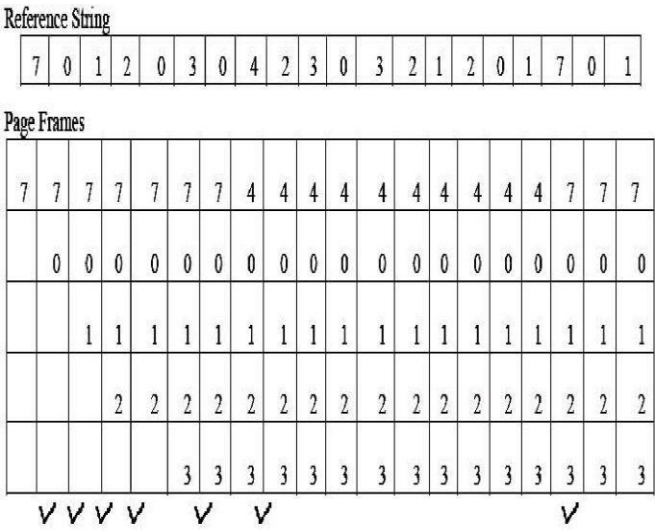


TABLE XII
FRAME-5 REPRESENTATION OF OPT

C. LRU

In LRU algorithm refrence page is replace with the recently used page.

Frame-1: 20 faults in a single memory-page.

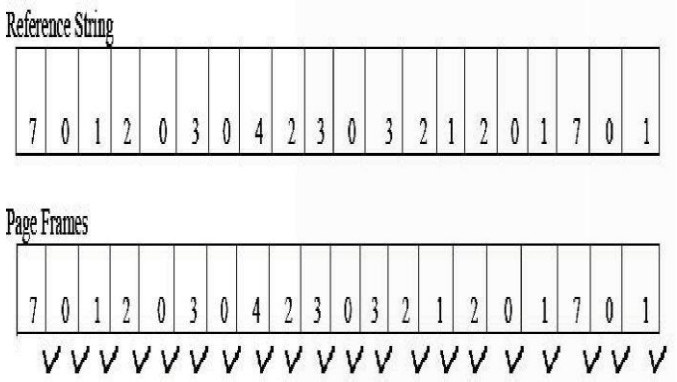


TABLE XIII
FRAME-1 REPRESENTATION OF LRU.

Frame-2: 17 faults in a two memory-page.

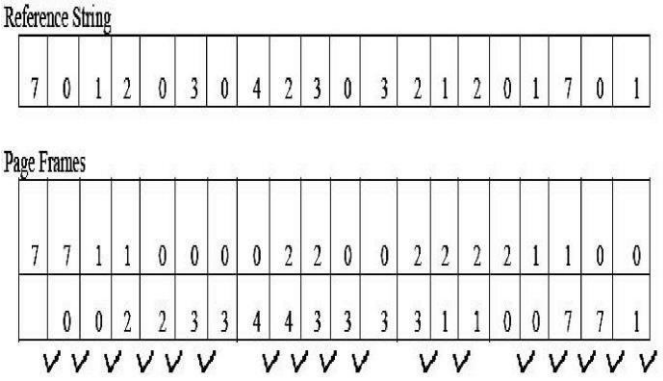


TABLE XIV
FRAME-2 REPRESENTATION OF LRU.

Frame-3: 12 faults in a three memory-page.

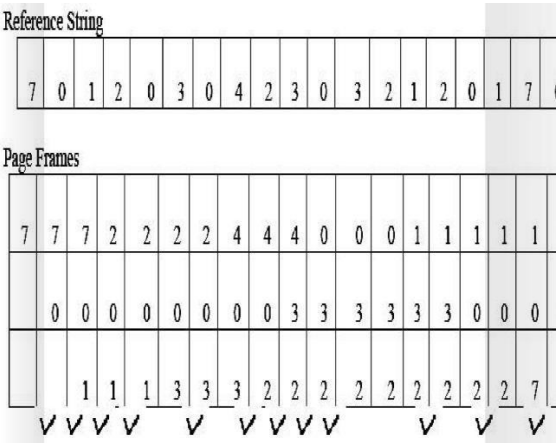


TABLE XV
FRAME-3 REPRESENTATION OF LRU.

Frame-4: 08 faults in a four memory-page.

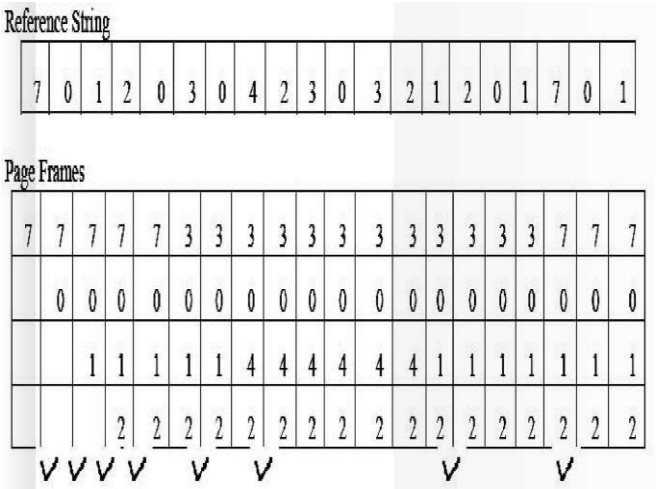


TABLE XVI
FRAME-4 REPRESENTATION OF LRU.

Frame-5: 07 faults in a five memory-page.

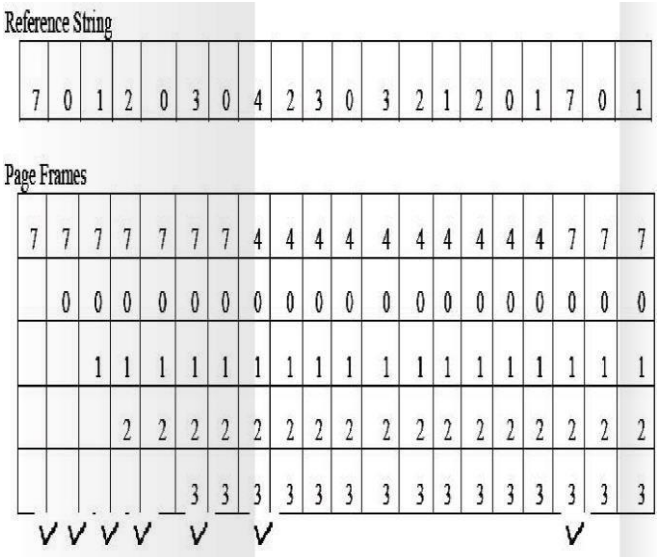


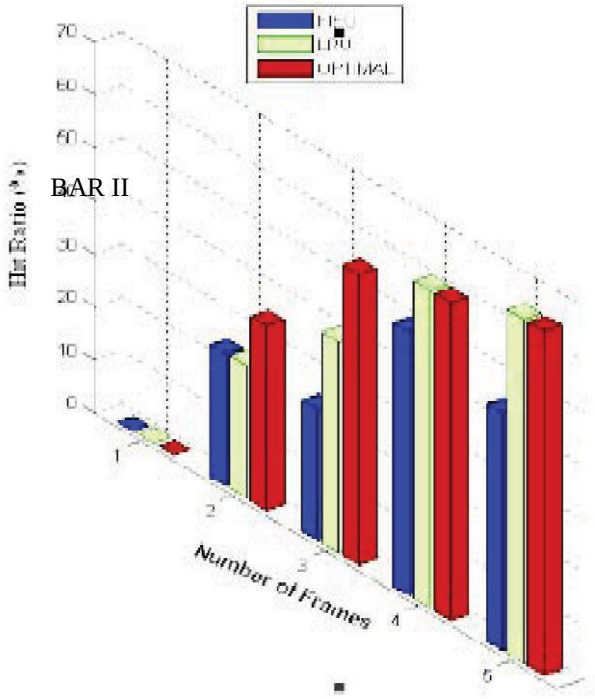
TABLE XVII
FRAME-5 REPRESENTATION OF LRU.

D. COMPARISON RESULTS

The experimental results are compare on the basis of page-faults and hit-ratio .We compare the results with the help of simulation tools and conclude the results which algorithm is better than the other algorithm. For this experiment we generate random sequences for FIFO,OP&LRU ,length of the randomly generated sequence is 20.In this experiment we generate five-cases for the memory allocation these are frame-1,frame-2,frame-3,frame-4,frame-5 and these results are shown in the table .Bar diagram is also shown below as a BAR II on the basis of these results.

Page Replacement Technique		Number of Frames				
		1-frame	2-frame	3-frame	4-frame	5-frame
FIFO	Page Fault	20	15	15	10	9
	Hit Ratio	0%	25%	25%	50%	45%
LRU	Page Fault	20	17	12	8	7
	Hit Ratio	0%	15%	40%	60%	65%
OPTIMAL	Page Fault	20	13	9	8	7
	Hit Ratio	0%	35%	55%	60%	65%

TABLE XVIII (COMPARISON OF FIFO,OPTIMAL&LRU.)



BAR II

COMPARISON OF FIFO,OPTIMAL&LRU.

Summary off the page replacement algorithms are shown in table below.

Algorithm	1 Comment
Optimal	Not implementable, but useful as a benchmark
NRU (Not Recently Used)	Very crude
FIFO (First-In, First-Out)	Might throw out important pages
Second chance	Big improvement over FIFO
Clock	Realistic
LRU (Least Recently Used)	Excellent, but difficult to implement exactly
NFU (Not Frequently Used)	Fairly crude approximation to LRU
Aging	Efficient algorithm that approximates LRU well
Working set	Somewhat expensive to implement
WSClock	Good efficient algorithm

TABLE XIX
SUMMARY OF PAGE REPLACEMENT ALGORITHMS.

V. CONCLUSION AND FUTURE WORK

This paper describes about the various types of page re-placement algorithms with respect to their page-faults and hit-ratio. On the behalf of results it's concluded that the LRU algo-rithm is the best as compare to the all other practical algorithm. Optimal algorithm is commonly used for the benchmark and the FIFO algorithm is endured with the belady's algorithm. When we increase the memory of the OS system the fault rate is also increased simultaneously. In this paper we study different types of page replacement algorithms and compare these algorithms on the basis of fault-

rate and hit-ratio and simulate the results with the help of simulation tools. The future work is also focused on LRU algorithm to make this algorithm more efficient with less fault rate.

REFERENCES

[1] F. S. Gharehchopogh, A. Maroufi, and I. Maleki, "Afrp: A new approach in page replacement algorithm with hybird aging and mamdani fuzzy interface algorithms," *International Journal of Academic Research*, vol. 6, no. 1, 2014.
[2] Y. Soumya and T. Ragunathan, "Lazy expression evaluation with de-mand paging in virtual memory management," *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 2, no. 1, pp. 1–3, 2012.
[3] W. Hong, "Study of page replacement algorithm based on experiment," *Advances in Biomedical Engineering*, vol. 10, p. 330, 2012.
[4] A. Khosrozadeh, S. Pashmforoush, A. Akbari, M. Bagheri, and N. Beikmahdavi, "Presenting a novel page replacement algorithm based on lru," *Journal of Basic and Applied Scientific Research*, vol. 2, no. 10,pp. 10 377–10 383, 2012.
[5] A. S. Tanenbaum, "Modern operating systems prentice hall," *Englewood Cliffs, NJ*, 1992.
[6] M. R. Gupta and S. Tokekar, "A novel pair of replacement algorithms on l1 and l2 cache for fft," *International Journal on Computer Science and Engineering*, vol. 2, no. 1, pp. 92–97, 2010.
[7] R. Gupta and S. Tokekar, "Proficient pair of replacement algorithms on l1 and l2 cache for merge sort," *arXiv preprint arXiv:1003.4088*, 2010.
[8] M. R. Hasan, M. S. Rahman, and C. S. Hyder, "Desynch lru: An efficient page replacement algorithm with desynchronized cache and ram," in *Proceedings of the 8th International Conference on Applied Informatics Eger, Hungary*, vol. 2, pp. 109–117.
[9] M. Sabeghi and M. H. Yaghmaee, "Using fuzzy logic to improve cache replacement decisions," *IJCSNS International Journal of Computer Science and Network Security, Seoul*, vol. 6, no. 3A, pp. 182–188, 2006.
[10] E. J. O'neil, P. E. O'Neil, and G. Weikum, "An optimality proof of the lru-k page replacement algorithm," *Journal of the ACM (JACM)*, vol. 46, no. 1, pp. 92–112, 1999.
[11] D. Lee, J. Choi, J.-H. Kim, S. H. Noh, S. L. Min, Y. Cho, and C. S. Kim, "Lrfu: A spectrum of policies that subsumes the least recently used and least frequently used policies," *IEEE transactions on Computers*, vol. 50, no. 12, pp. 1352–1361, 2001.
[12] Janapsatya, A. Ignjatovic´, J. Peddersen, and S. Parameswaran, "Dueling clock: Adaptive cache replacement policy based on the clock algorithm," in *Proceedings of the Conference on Design Automation and Test in Europe*. European Design and Automation Association, 2010, pp. 920–925.
[13] S. Jiang and X. Zhang, "Making lru friendly to weak locality workloads: A novel replacement algorithm to improve buffer cache performance," *IEEE Transactions on Computers*, vol. 54, no. 8, pp. 939–952, 2005.
[14] S. Jiang, F. Chen, and X. Zhang, "Clock-pro: An effective improvement of the clock replacement." in *USENIX Annual Technical Conference, General Track*, 2005, pp. 323–336.
[15] S. S. Daula, K. S. Murthy, and G. A. Khan, "A throughput analysis on page replacement algorithms in cache memory management,"
system, vol. 1, p. 12, 2012.