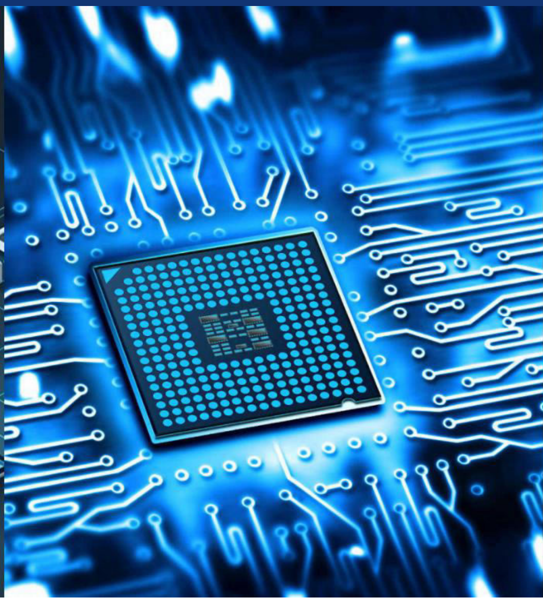
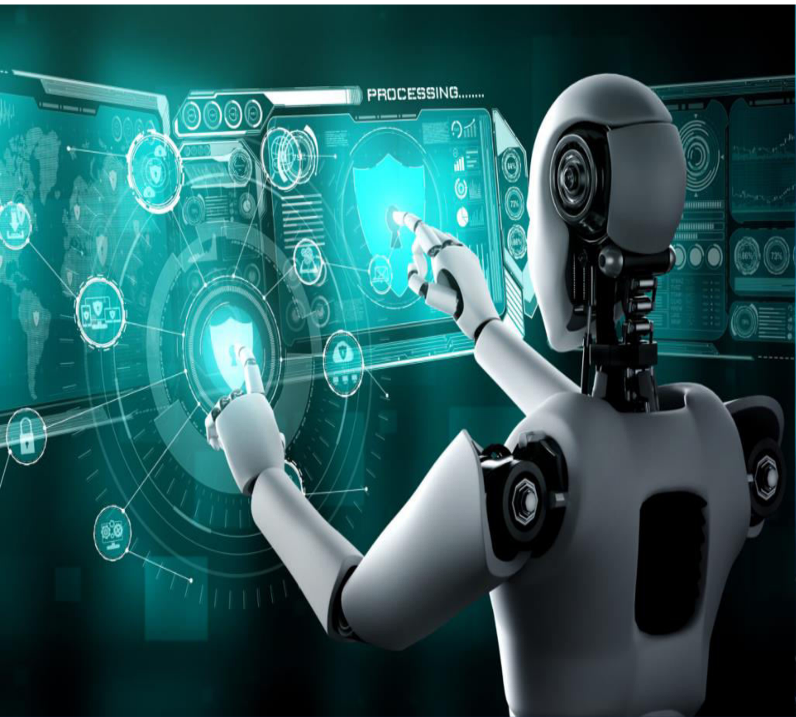
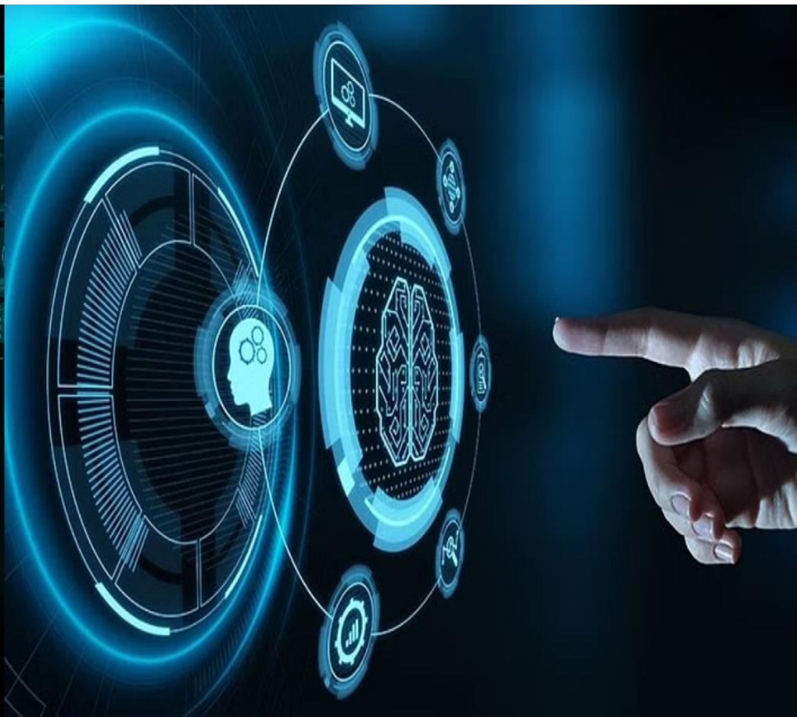
ISSN(O): 2320-9801
ISSN(P): 2320-9798


International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)





Impact Factor: 8.771

Volume 14, Issue 4, April 2026

© 2026 IJIRCCE | Volume 14, Issue 4, April 2026 DOI: 10.15680/IJIRCCE.2026.1404140

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

LPFIFO Page Replacement Policy for Wireless Systems and Chip Multiprocessor Architectures

Ananthoju Vijay Kumar[1], Preethi Tolaguppala[2], Jangam Pallavi[2]

Associate Professor, Department of Computer Science and Engineering, School of Computer Science and Engineering, Jain University, Bangalore, India[1]

B. Tech CSE 3rd Year Student, Department of Computer Science and Engineering, School of Computer Science and Engineering, Jain University, Bangalore, India[2]

ABSTRACT: Cache memory plays a critical role in improving the performance of real-time systems that require rapid data access. Traditional Cache replacement policies such as First-In-First-Out (FIFO), Least Recently Used (LRU), Most-Recently-Used (MRU) and Optimal Page Replacement (OPR) are often insufficient in wireless systems and chip multi-processor (CMP) architectures due to their static decision strategies. This paper proposed a cache replacement policy named LPFIFO (LEAST PAGE FAULT FIFO). The core strength of LPFIFO lies in prioritizing larger pages first for eviction decision process. The proposed model is evaluated through simulation across a variety of realistic access patterns and benchmark dataset, comparing its performance against Belady's optimal algorithm, FIFO, LRU and MRU algorithm. Experimental results show that LPFIFO achieves improvement in cache hit rate over conventional strategies and performs competitively with Belady's optimal policy in several scenarios. These results demonstrate LPFIFO's potential as a viable replacement policy for future generation cache architectures in CMP and mobile-age environments.

KEYWORDS: Belady's Algorithm, Cache Memory, Cache Replacement Policy, Eviction Strategy, FIFO, LPFIFO, LRU, MRU, Optimal Page Replacement (OPR), Page Size and Real-Time Systems.

I. INTRODUCTION

Caching is a fundamental technique in digital computer architecture, intended to bridge the performance gap between high-speed processors and comparatively slower main memory. Among the many factors influencing cache effectiveness, the cache replacement policy plays an essential role, as it determines which data blocks are dispossessed when the cache reaches full capacity. Efficient replacement policies are vital for optimizing system performance, particularly in Coherent Chip Multiprocessor (CMP) systems and wireless computing environments, where workload dynamics and data access patterns are highly variable.

Traditional strategies such as First-In-First-Out (FIFO), Least Recently Used (LRU), and their variants, while generally adopted, often fall short in these scenarios due to their inability to adapt to changes in data relevance and size. To address these limitations, this work introduces a novel cache replacement policy named Least Page Fault First in First Out (LPFIFO).

LPFIFO extends conventional age-based replacement strategies by considering the page size into the eviction decision making process. The core objective is to prioritize the eviction of larger pages which are entered first in to the cache memory, there by preserving the smaller and which are entered recently that contributes more efficiently to performance. The dual-parameter approach enables a more nuanced and adaptive cache management mechanism, striking a balance between recency of use and data utility.

The LPFIFO policy is evaluated through extensive simulation across diverse and realistic memory access patterns. Its performance is benchmarked against well-established policies such as LRU, FIFO, and the theoretically optimal Belady's algorithm. Experimental results demonstrate that LPFIFO achieves significant improvements in cache hit rates, with measurable reductions in memory access overhead. These findings highlight LPFIFO's potential as a

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

scalable and efficient replacement policy for next-generation cache architectures in CMPs, wireless systems, and mobile-edge computing environments.

II. RELATED WORK

Conventional replacement policies such as Least Recently Used (LRU) [1], First-In-First-Out (FIFO) [2], and Most Recently Used (MRU) [3] have been extensively studied and widely implemented in general-purpose systems. These strategies are primarily based on temporal access patterns**,** assuming that recently or frequently accessed pages are more likely to be reused. However, they often ignore the context or characteristics of the data being cached.

The LRU policy operates on the principle that recently accessed data is more likely to be reused, making it intuitive and effective in certain workloads. However, it introduces implementation overhead due to frequent list updates or stack maintenance. In contrast, MRU evicts the most recently accessed page first. While it may be suitable for specific real-time applications, it performs poorly in workloads with strong temporal locality**.**

The FIFO algorithm, due to its simplicity, evicts the oldest page in the cache regardless of how frequently or recently it has been accessed. This leads to situations where frequently used data may be prematurely removed, resulting in suboptimal hit ratios.

Among all, the Belady's Optimal Page Replacement Policy is considered the theoretically most efficient**.** It avoids evicting pages that will be used in the near future. However, since it relies on perfect future knowledge**,** its implementation is not feasible in real-world systems. It is mainly used as a benchmark to compare the performance of practical algorithms.

To overcome the limitations of classical approaches, adaptive and frequency-based strategies like Adaptive Replacement Cache (ARC) [4] have been proposed. ARC dynamically balances between recency and frequency but often introduces increased algorithmic complexity and may not scale well with varying page sizes or workload patterns**.**

In modern computing environments such as Mobile Edge Computing and Multiprocessor Systems**,** cache access patterns tend to be more bursty and unpredictable, which further limits the effectiveness of traditional temporal-based algorithms [5].

Advanced strategies like the Quad-Age and MLP-Aware Policies have been introduced to address these challenges. These methods incorporate recency, frequency, and future access prediction**,** aiming to enhance decision-making in multi-core systems**.** The MLP-aware policy in particular focuses on improving Memory-Level Parallelism (MLP) by optimizing access overlap, leading to better system throughput.

Despite numerous efforts to balance simplicity and efficiency in page replacement strategies, challenges remain in adapting to loop-intensive and sequential access patterns without increasing algorithmic complexity. To address these limitations, this work proposes a lightweight yet effective strategy, LPFIFO, which aims to improve page retention and fault reduction in such patterns while preserving FIFO's simplicity.

III. PROPOSED ALGORITHM

A. *Design Considerations:*
The **enhanced First-In-First-Out page replacement algorithm**, which is more efficient than Belady's algorithm in certain scenarios, is based on the principle that **pages with larger size and which are entered first should be evicted first**. Due to this principle, the algorithm is also known as Larger Page First in First Out **(LPFIFO)**. The eviction of the pages is done on the basis of a combined score calculated from the following formulae.
Let:
S_i =Size of page i
E_i= Eviction score of page i
Then, eviction score is calculated as:
$E_i = Si + C$

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|
International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)
(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Where C is the constant used for improving the algorithm performance.
The page with the highest E_i is selected for eviction.

B. *Description of the Proposed Algorithm:*

Step1: Computed Page Size

For every new page, its memory size (in KB/MB) should be computed and recorded. This value decides whether it should be there in cache or to be evicted on the cache miss situation.
Let S_{new} is memory size of new page (in KB or MB)
C_{max} is total cache size available
C_{used} is the total size currently used in the cache

A) Compute total cache available
$$C_{avail} = C_{max} - C_{used}$$
B) Check if new page fits
$S_{new} <= C_{avail}$
If true insert the page
Otherwise evict one or more pages until enough space is available
C) Update cache used and available cache
$C_{used} = C_{used} + S_{new}$
$C_{avail} = C_{max} - C_{used}$

Step 2: Compute Eviction Score

An eviction score is calculated for each page using the following formula:
$$\boldsymbol{Eviction\ Score(i) = Page\ Size(i) + C}$$
Where C is the constant used for tuning the performance of the algorithm.

Step 3: Handle Page Faults

On each page fault, the largest page which has entered cache first will be removed, and this deletion of pages will be performed till the sufficient free space created in the cache.
Let
P={ p1,p2,…pn } be the set of pages currently in the cache
Si is size of the page pi
E_i is eviction score of page pi
C_{used} is the cache used
C_{max} is the cache total storage capacity
C_{avail} is the available free space in the cache
Ei = Si
S_{new} = size of P_{new}
The page to be evicted on a page fault is P_k
S_k is the size of the page P_K to be evicted from the cache
While(($C_{avail} <= S_{new}$) && Page Fault)
{
$E_k = max(Si)$ where 1<=I <=n
Evict the page P_k
$C_{avail} = C_{avail} - S_k$
}

Step 4: Insert New Page & Update Counters

The new page is added to the cache.
Let P = { p1,p2,..pn } Current set of pages in the cache
P_{new} is new page to be inserted
a) Insert new page
Add P_{new}
b) Replacement cycle

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Repeat eviction and insertion:
While page fault occurs, repeat:
Repeat : evaluate Ei =Si
Evict page with max(Ei)

IV. SIMULATION RESULTS

To conduct the experiment and analyze the performance of the proposed algorithm over other traditional cache replacement algorithms, we considered the following factors, frame size, page reference patterns and page faults across different policies. In the nine different scenarios the experiment has been conducted and the results are analyzed properly to understand the performance of the proposed policy. The test cases are designed with frame size varied from 3 to 9, page reference patterns with sequential, loop-based and mixed. The metrics evaluated is page faults across different policies.

Case 1: Random Short Burst Pattern
In the present scenario, page reference string follows a random short burst pattern, and the cache frame size is limited to three pages. The reference string assumed for the experiment is as follows,
Page References: 7,0,1,2,0,3,0,4
Corresponding Page Sizes: 10,1,4,5,1,4,1,9

Regardless of the page reference string pattern and frame size, the proposed cache replacement policy exhibited performance comparable to conventional algorithms, such as LRU,MRU and optimal page replacement algorithm. Whereas the proposed algorithm performance is little better than the FIFO performance.In the present scenario, the model performance is with 2 catch hits, resulting in a cache hit ratio of 0.25.

The present results shows the model's capacity to maintain competitive performance under random access patterns and variety of things page sizes, ensuring competent cache utilization yet in constrained environments.
The number of page faults for each algorithm is as follows:

Algorithm	FIFO	LPFIFO	LRU	MRU	Belady's
Page Faults	7	6	6	6	6
Cache Hit Ratio	0.125	0.25	0.25	0.25	0.25

Table1. Page Faults of different algorithms under comparison with mixed access pattern

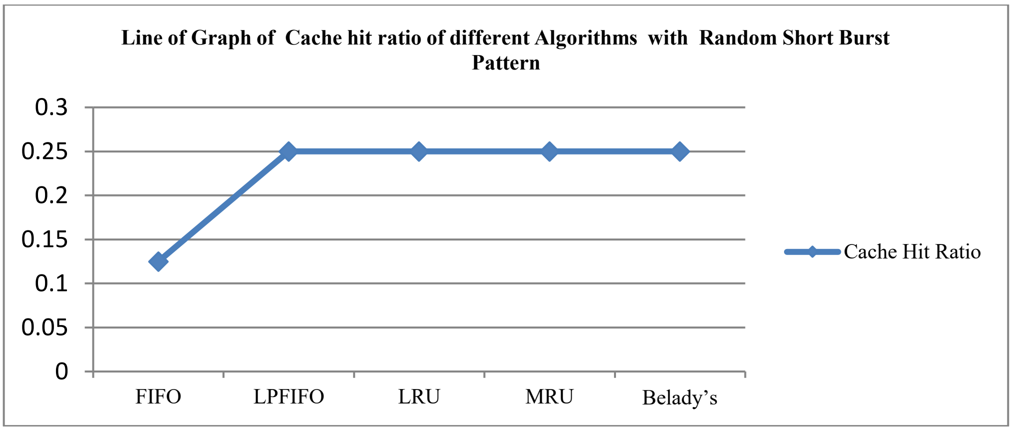


Figure1. Line of Graph of Cache hit ratio of different Algorithms with Random Short Burst Pattern.

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Case 2: Sequential Stream Pattern

In the present scenario, the page reference pattern follows a sequential access, and the cache frame size is limited to three pages. The reference string considered for evaluation is,

Page References: 7,1,2,3,4,0,0,0

Corresponding Page Sizes: 10,4,5,4,9,1,1,1

Despite the variations in page sizes and limited frame capacity, the proposed cache replacement model demonstrates performance comparable to traditional algorithms such as FIFO and optimal page replacement algorithms and better than LRU and MRU. In the assumed scenario, the model successfully achieved 2 cache hits, resulting in a cache hit ratio of 0.25.

This outcome highlights the model's capability to maintain competitive performance under sequential access patterns and dynamic page sizes, ensuring efficient cache utilization even in constrained environments.

Algorithm	FIFO	LPFIFO	LRU	MRU	Belady's
Page Faults	**6**	**6**	7	7	**6**
Cache Hit Ratio	**0.25**	**0.25**	0.125	0.125	**0.25**

Table2. Page faults and Cache hit ratio of different algorithms under comparison with Sequential Pattern.

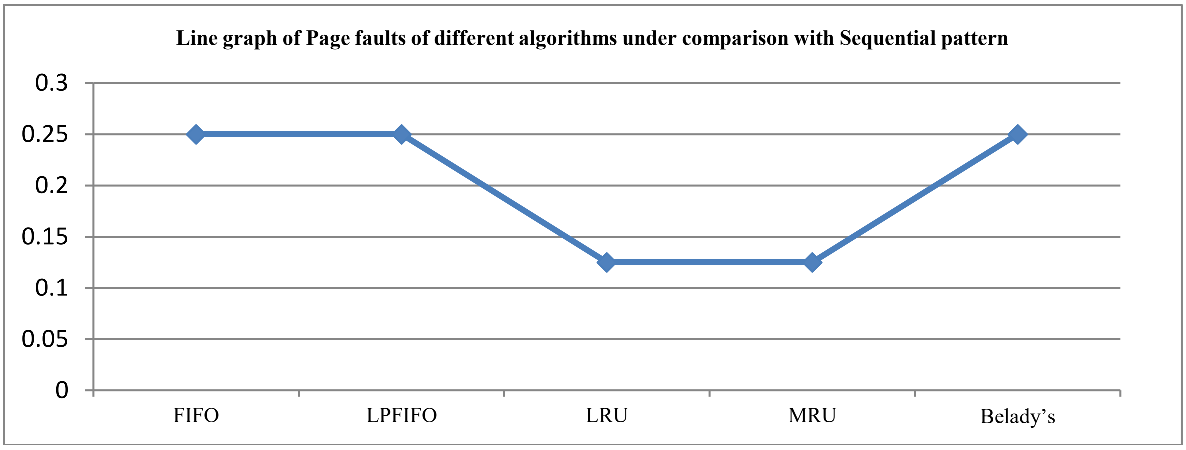


Figure2. Line graph of Page faults of different algorithms under comparison with Sequential pattern.

Case 3: Repeating Working Set Pattern

In the case 3, the page reference pattern set follows repeating working set pattern, which is exclusively designed to test the performance of the LPFIFO in this kind of scenarios, where the set of page references would be repeated. Here, the catch frame size is limited to three pages as similar to the previous case. The page reference string considered for evaluation is,

Page References: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

Corresponding Page Sizes: 4,6,2,3,4,6,1,4,6,2,3,1.

With the repeating nature of the page references and the limited frame capacity, the futuristic cache replacement model **LPFIFO** demonstrated performance equal to that of the **Belady's Optimal Page Replacement Algorithm** and performed far better than the other conventional algorithms.

Algorithm	**FIFO**	**LPFIFO**	**LRU**	**MRU**	**BELADY**
Page Faults	10	7	10	10	7
Cache Hit Ratio	0.17	**0.42**	0.17	0.17	0.42

Table3. Cache Hit Ratio and Page faults of different algorithms under comparison with Loop and replacement test.

© 2026 IJIRCCE | Volume 14, Issue 4, April 2026 DOI: 10.15680/IJIRCCE.2026.1404140

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

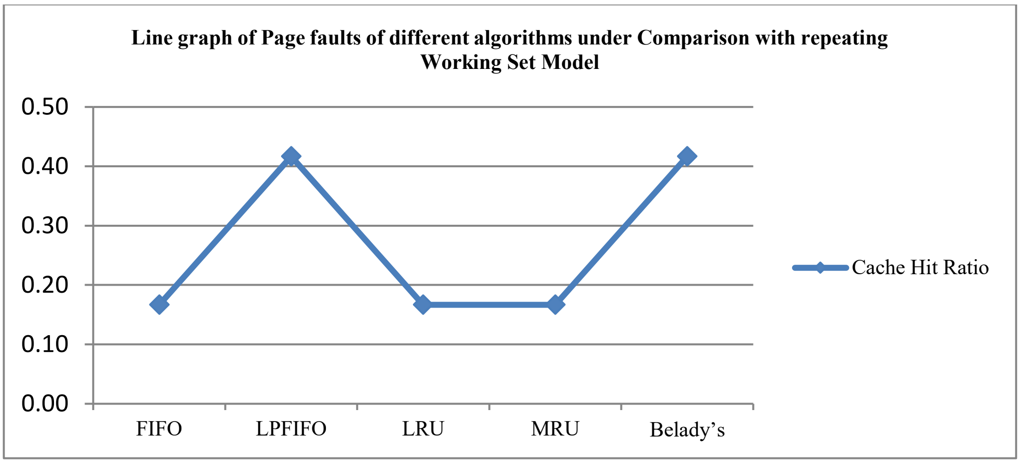


Figure3. Line graph of Page faults of different algorithms under Comparison with repeating Working Set Model.

Case 4: Temporal Locality in Mid Length Sequence

To test the performance of LPFIFO with the traditional cache replacement policies under repetitive access in mid-sized memory here experiment has been conducted. In this case page references were 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1, 7 and frame size was 5. The series contains repeated access of pages combined with continuous re-uses of recently accessed pages which enables effective testing of memory retention across time. LPFIFO maintains a better retention rate by factoring space. The performance of LPFIFO is equals to the Belady's performance. The proposed cache replacement policy cache hit ratio was 0.7.

Algorithm	FIFO	LPFIFO	LRU	MRU	Belady's
Page Faults	9	7	9	7	7
Catch Hit Ratio	0.57	0.67	0.57	0.67	0.67

Table4. Cache Hit Ratio and Page faults of different algorithms under Temporal Locality in Mid Length Sequence.

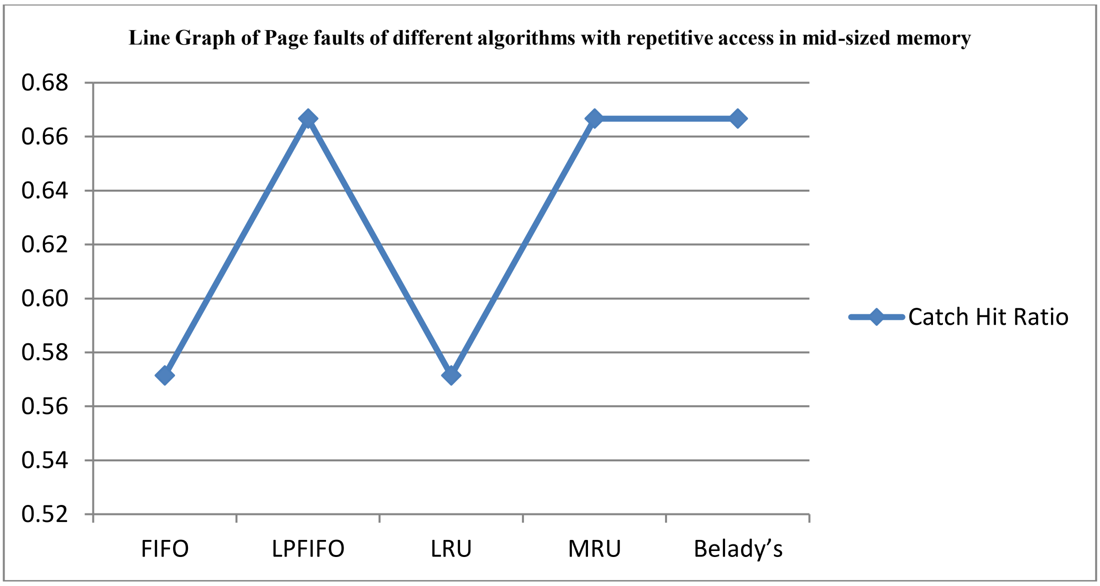


Figure 4. Line Graph of Page faults of different algorithms with repetitive access in mid-sized memory.

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

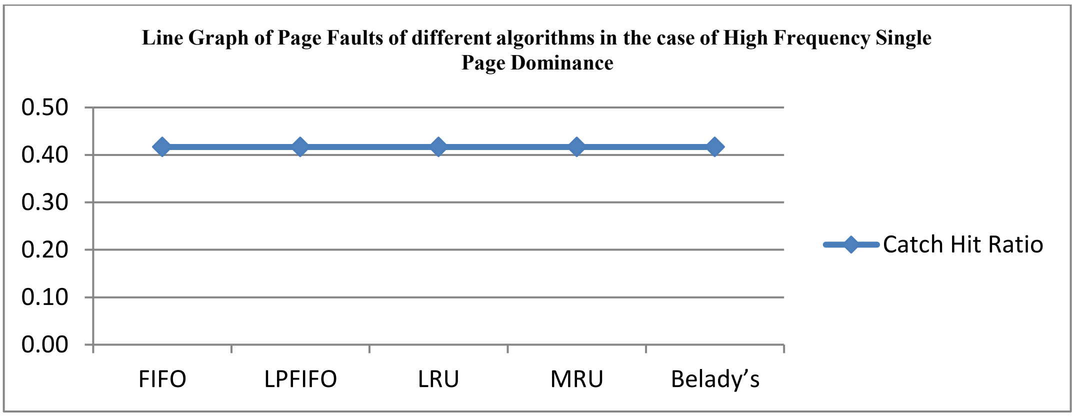
Case 5: High Frequency Single Page Dominance

When a lengthy string of page references with single page dominance were considered for the study, we have received the following results. There is a clear dominance of LPFIFI and belady's optimal page replacement algorithm over the other three policies. In this case there were 21 pages in the page reference string, and page 0 was referred more number of times, i.e. 6 times. Here, LPFIFO and Belady algorithm cache hit rate was 25% which is better than LRU and MRU policies. The page reference string was 7,0,0,0,0,0,0,1,2,3,4,2,3,3,2,1,2,1,7,1,7 and page sizes were 9,2,2,2,2,2,2,3,4,5,7,3,4,4,5,2,4,3,9,3,9. The proposed algorithm catch hit ratio was 0.7

The number of page faults for each algorithm is as follows.

FIFO	LPFIFO	LRU	MRU	BELADY'S
Page Faults	9	9	9	9
Catch Hit Ratio	0.42	0.42	0.42	0.42

Table 5. Page Faults of different algorithms in the case of High Frequency Single Page Dominance



Case 6: Balanced Long Reference Pattern

To test the performance of the proposed cache replacement policy here a balanced long page reference pattern was used, the length of the page reference string was 35, frame was divided into 9 pages and their page sizes were 2,3,5,6,2,3,7,2,3,5,6,7,2,3,5,6,2,10,8,9,10,10,8,7,2,3,5,6,2,3,8,2,3,10,6. When the experiment was conducted the proposed algorithm performance was much better than the traditional policies. The experiment results were as follows. The proposed algorithm cache hit ratio was 0.77.

Algorithm	FIFO	LPFIFO	LRU	MRU	Belady's
Page Faults	9	8	9	9	9
Catch Hit Ratio	0.46	0.77	0.46	0.46	0.46

Table6. Page faults of different algorithms with Large-Scale reference pattern.

© 2026 IJIRCCE | Volume 14, Issue 4, April 2026 DOI: 10.15680/IJIRCCE.2026.1404140
www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|
International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)
(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

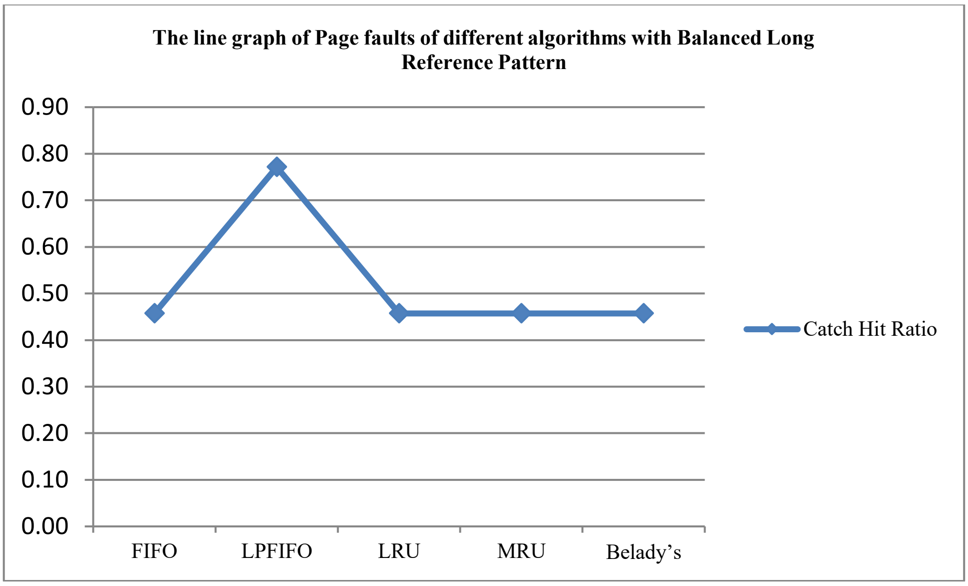


Figure6. The line graph of Page faults of different algorithms with Balanced Long Reference Pattern.

Case 7: Reinforced Repetition Pattern

To cross verify the performance of the LPFIFO again we have repeated some pattern of the page references this time, while the page reference string and frame size remained same. The page reference string was 1,2,2,2,2,2,2,2,3,4,1,5,1,3,4,5,1,3,4,1,8,6,7,9,8,6,5,1,3,4,1,6,1,9 and the pages size were 2,3,5,6,2,3,7,2,3,5,6,7,2,3,5,6,2,10,8,9,10,10,8,7,3,5,6,2,3,8,2,3,10,6. The proposed algorithm cache hit ratio was a bit more than the other algorithms.

	FIFO	LPFIFO	LRU	MRU	BELADY'S ANOMALY
Page Faults	9	7	9	9	9
Catch Hit Ratio	0.74	0.8	0.74	0.74	0.74

Table7. Page faults of different algorithms with reinforced repetition pattern.

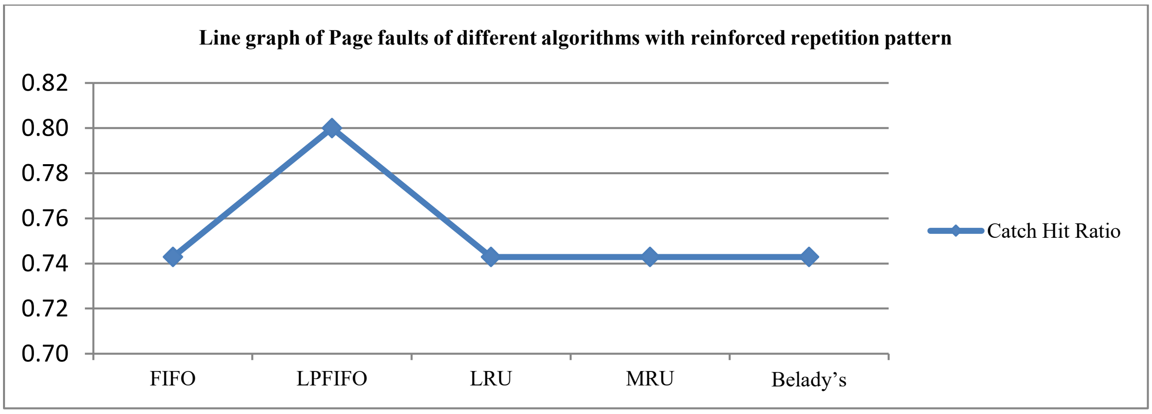


Figure7. Line graph of Page faults of different algorithms with reinforced repetition pattern.

© 2026 IJIRCCE | Volume 14, Issue 4, April 2026 DOI: 10.15680/IJIRCCE.2026.1404140

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Case 8: Stress Test on Limited Frame Size

Here the proposed algorithm performance has been compared with page reference string length of 35 and 3 page frames have been used. The page reference string was 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5, 1, 2, 3, 4, 1, 8, 6, 7, 9, 8, 6, 5, 1, 2, 3, 4, 1, 2, 6, 1, 2, 9, 4 and page sizes were 2,3,5,6,2,3,7,2,3,5,6,7,2,3,5,6,2,10,8,9,10,10,8,7,2,3,5,6,2,3,8,2,3,10,6. In this case the performance of LPFIFO was much better than FIFO, LRU, MRU and Belady's algorithm. The number of page faults for each algorithm is as follows. The LPFIFO algorithm catch hit ratio was 0.7.

Algorithm	FIFO	LPFIFO	LRU	MRU	BELADY'S
Page Faults	31	22	31	25	22
Catch Hit Ratio	0.1	0.4	0.1	0.3	0.4

Table8. Page faults of different algorithms with Stress Test on Limited Frame Size.

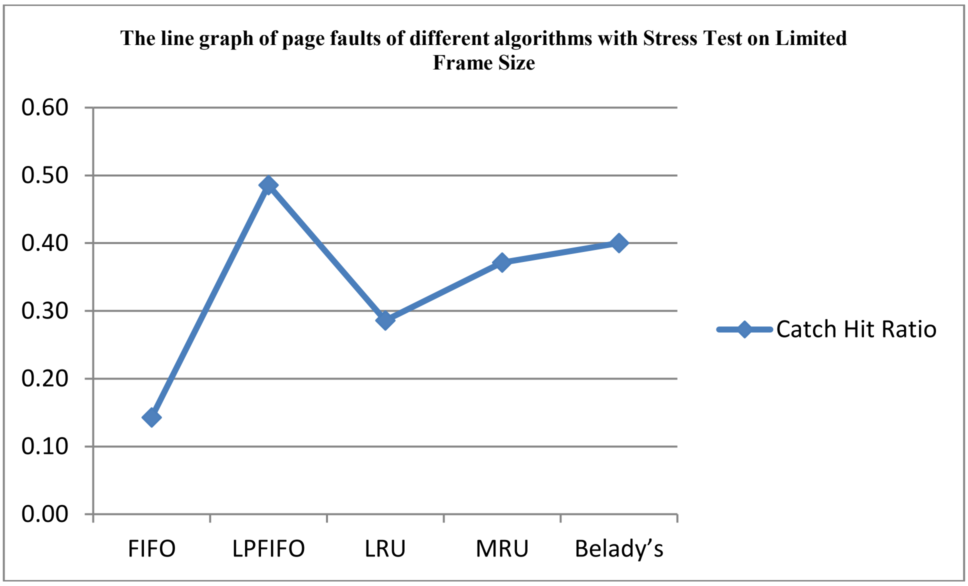


Figure8. The line graph of page faults of different algorithms with Stress Test on Limited Frame Size.

Case 9: FIFO favoured repetition pattern 35/3

To test the performance of the proposed model on very short frame with the same memory reference string, here we did the experiment. The page reference string was 1,2,2,2,2,2,2,2,2,3,4,1,5,1,3,4,5,1,3,4,1,8,6,5,1,3,4,1,6,1,9 and their corresponding page sizes were 1,7,7,7,7,7,7,7,7,1,2,1,2,3,2,6,5,4,6,5,6,2,3,1,2,2,3,2,6,3,3,6,3,7,2,4. The frame was capable of accommodating 3 pages at a time. In this case LPFIFO and optimal page replacement algorithms performed equal and much better than the remaining all cache replacement algorithms. The proposed algorithm cache hit ratio was 0.54.

	FIFO	LPFIFO	LRU	MRU	BELADY'S
Page Faults	22	16	27	25	16
Catch Hit Ratio	0.37	0.54	0.23	0.29	0.54

Table9. Page faults of different page replacement algorithms with FIFO favoured repetition pattern.

© 2026 IJIRCCE | Volume 14, Issue 4, April 2026 DOI: 10.15680/IJIRCCE.2026.1404140

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

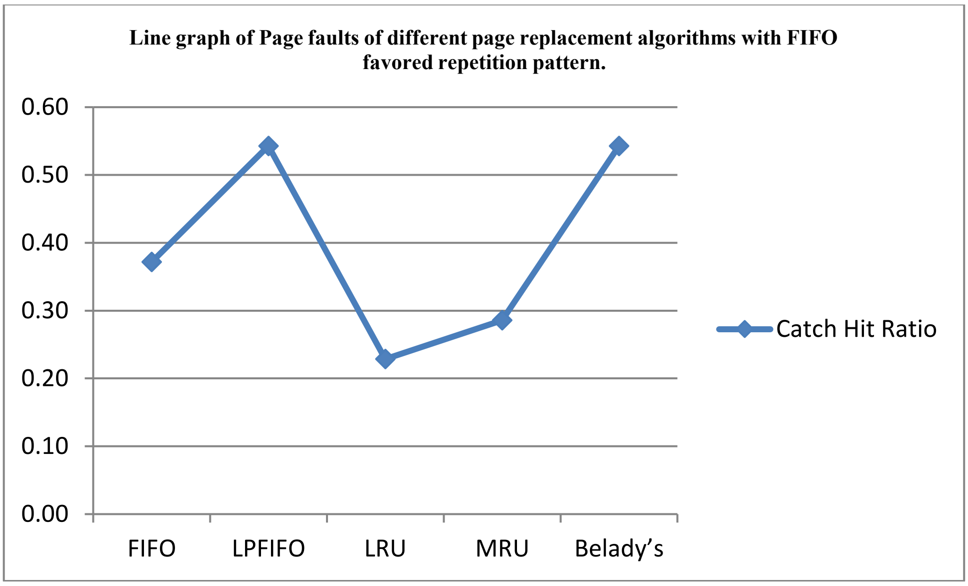


Figure9. Line graph of Page faults of different page replacement algorithms with FIFO favoured repetition pattern.

VI. CONCLUSION AND FUTURE WORK

The simulation results showed that the proposed algorithm performs better with the total transmission energy metric than the maximum number of hops metric. The proposed algorithm provides energy efficient path for data transmission and maximizes the lifetime of entire network.As the performance of the proposed algorithm is analyzed between two metrics in future with some modifications in design considerations the performance of the proposed algorithm can be compared with other energy efficient algorithm. We have used very small network of 5 nodes, as number of nodes increases the complexity will increase. We can increase the number of nodes and analyze the performance.

REFERENCES

1. AnjumAsmaand GihanNagib,'Energy Efficient Routing Algorithms for Mobile Ad Hoc Networks–A Survey', International Journal of Emerging Trends & Technology in computer Science, Vol.3, Issue 1, pp. 218-223, 2012.
2. Hong-ryeol Gil1, Joon Yoo1 and Jong-won Lee2,'An On-demand Energy-efficient Routing Algorithm for Wireless Ad hoc Networks', Proceedings of the 2nd International Conference on Human. Society and Internet HSI'03, pp. 302-311, 2003.
3. S.K. Dhurandher, S. Misra, M.S. Obaidat, V. Basal, P. Singh and V. Punia,'An Energy-Efficient OnDemand Routing algorithm for Mobile Ad-Hoc Networks', 15 th International conference on Electronics, Circuits and Systems, pp. 958-9618, 2008.
4. DilipKumar S. M. and Vijaya Kumar B. P. ,'Energy-Aware Multicast Routing in MANETs: A Genetic Algorithm Approach', *International Journal of*Computer *Science and Information Security* (IJCSIS), Vol. 2,2009.
5. AlGabriMalek, Chunlin LI, Z. Yang, NajiHasan.A.H and X.Zhang,' Improved the Energy of Ad hoc On- Demand Distance Vector Routing Protocol', International Conference on Future Computer Supported Education, Published by Elsevier, IERI, pp. 355-361, 2012.
6. D.Shama and A.kush,'GPS Enabled E Energy Efficient Routing for Manet', International Journal of Computer Networks (IJCN), Vol.3, Issue 3, pp. 159-166, 2011.
7. Shilpajain and Sourabhjain,'Energy Efficient Maximum Lifetime Ad-Hoc Routing (EEMLAR)', international Journal of Computer Networks and Wireless Communications, Vol.2, Issue 4, pp. 450-455, 2012.
8. Vadivel, R and V. MuraliBhaskaran,'Energy Efficient with Secured Reliable Routing Protocol (EESRRP) for Mobile Ad-Hoc Networks', Procedia Technology 4,pp. 703- 707, 2012.
9. Nobuo Ezaki, Marius Bulacu Lambert , Schomaker , "Text Detection from Natural Scene Images: Towards a System for Visually Impaired Persons" , Proc. of 17th Int. Conf. on Pattern Recognition (ICPR), IEEE Computer Society, pp. 683-686, vol. II, 2004

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

10. Mr. Rajesh H. Davda1, Mr. Noor Mohammed, " Text Detection, Removal and Region Filling Using Image Inpainting", International Journal of Futuristic Science Engineering and Technology, vol. 1 Issue 2, ISSN 2320 – 4486, 2013

11. Uday Modha, Preeti Dave, " Image Inpainting-Automatic Detection and Removal of Text From Images", International Journal of Engineering Research and Applications (IJERA), ISSN: 2248-9622 Vol. 2, Issue 2, 2012.

12. Muthukumar S, Dr.Krishnan .N, Pasupathi.P, Deepa. S, "Analysis of Image Inpainting Techniques with Exemplar, Poisson, Successive Elimination and 8 Pixel Neighborhood Methods", International Journal of Computer Applications (0975 – 8887), Volume 9, No.11, 2010

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING