International Interdisciplinary Virtual Conference on
'Recent Advancements in Computer Science, Management and Information Technology'
International Journal of Scientific Research in Computer Science,
Engineering and Information Technology| ISSN : 2456-3307 (www.ijsrcseit.com)

Comparative Analysis of a Novel Web Caching Algorithm CachoHit with LRU and FIFO

Dr. Anil Dudhe
Department of Computer Science, Phulsing Naik Mahavidyalaya, Pusad, Maharashtra, India

ABSTRACT
In order to improve the network bandwidth and the server's reliability, availability and response time [1] Web caching is a very important feature. A key factor behind a cache [2] is its page replacement policy, which decides which page will be evicted from the cache to make space for a new page. In this paper we have made comparative analysis of Web caching algorithms in optimizing the overall performance (based on important measures such as network traffic, hit ratio, response time, etc).
Keywords: Caching, LRU, FIFO, LFU, page replacement algorithm

I. INTRODUCTION

Everyday Internet traffic is increasing heavily as a result showing its impact on network congestion which increases server load ultimately affecting access time, response time and many other factors. Thus, web caching [3] is crucial for reducing the load on network, shorten network latency and improve clients' waiting time. There are n number of web caching policies like, First In First Out, Least Recently Used, Least Frequently used, Optimal Page Replacement [4] and so on are implemented which decides how and which object or page should be evicted from cache to make space available for incoming request, Most of the existed Web cache replacement algorithms [5] consider few factors and ignore many other factors that have an impact on the efficiency of the Web caching. Moreover, a combination of these factors to get wise replacement decision is not a simple task. Therefore, many objects, which are stored in the cache, are never requested again or for a long time. This leads to *"cache pollution"*, meaning is that objects are inactive objects. This causes a reduction of the effective cache size and negatively affects the performance of Web proxy caching [6].
Many researchers have proven disadvantages of existed conventional algorithms and tried to overcome these but still there are some challenges which can be resolved by various techniques. In this paper we are comparing of Web caching algorithms in optimizing the overall performance based on hit ratio.

WORKING OF CACHE

A web cache is a methodology for the temporary storage (caching) of web pages, such as HTML pages and images to optimize the use of bandwidth, server load and perceived lag [7]. A web cache stores copies of documents passing through it; subsequent requests may be satisfied from the cache if certain conditions are

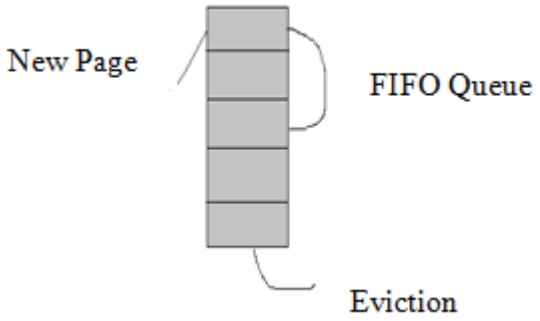
met. Web sites are continually updating their contents [8]. News headlines change, stock quotes change, weather changes. It may seem that caching is not worthwhile if it is returning outdated material. A traffic report that is two hours old doesn't do you much good. Fortunately there are checks and balances in place to ensure that the content you are viewing is current. Web sites are made up of many small pieces that come together to make a complete page. A site might have logos, photographs, tables, text, and sounds. Each item will be cached as a different object, and some items may not cache at all. For example, when you access CNN.com frequently your cache may hang on to the CNN logo objects some advertising bars, and the rest of the stuff that makes up the basic look of the CNN Web site[9]. But the news items will not sit in cache because they change so often. In this case your cache has made the CNN site much easier and faster to download because all the static graphics are already on hand and the only thing you need to complete the picture is the news content.

II. CACHE REPLACEMENT POLICIES: IN BRIEF

Caching, an essential approach in present computing, extensively used in application in storage systems, databases, Web servers, processors, file systems, disk drives, operating systems, and data compression applications. Cache replacement policy plays a vital role in web caching. Web Caching can enhance data access response time by preserving data that tend to be requested in the future. Therefore, the design of efficient cache replacement algorithms is very important. Due to limitation on size of Cache, a cache replacement policy is needed to handle the cache content carefully. When the cache to store the current/incoming object, the replacement policy will determine which object is to be evicted to make space for the new object. The optimal policy improve cache hit rates, and to reduce loads on the server. The following are different page replacement policies for review:

2.1 Least Recently Used (LRU)

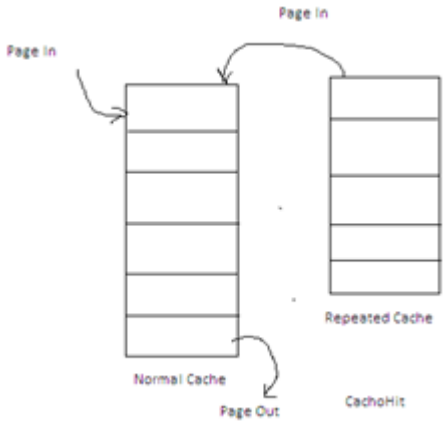
This is most commonly used page replacement caching algorithm which keeps recently used request close to the top of the cache. Whenever a new request is accessed, the LRU places it at the top of the cache. And whenever the cache limit has been reached, the request that have been accessed less recently will be evicted from the bottom of the cache. This can be a costly as it has to maintain the access time-stamp of request. In addition to that, when a LRU cache algorithm deletes an less recently request, the access time-stamp must be changed on all other request also.



2.2 First In First Out (FIFO)

According to eviction policy of FIFO page entered is evicted in the same order as they come in. In FIFO when new request is made and consider cache is reached at the max limit, the page that was placed first (First-In) in the store is the consider for eviction (First-Out).

2.3 CachoHit



III. IMPLEMENTATION

The performance of above discussed caching algorithms is evaluated on various parameters. The most important factor is cache hit ratio, which depends on the page replacement policy.

Cache Hit Ratio:

The Cache Hit Ratio is the ratio of the number of cache entries are repeated to the number of request available, usually expressed as a percentage. Depending on cache size configured, expected hit ratios may vary from 60% to greater than 99% [10].

To calculate hit ratio we have conducted test from on various data sets. The cache size was from 400 MB random requests and 900 MB size.

Page hit ration is calculated using following formula:

$$\text{Cache hit ratio} = [\text{Cache Hits} / (\text{Cache Hits} + \text{Cache Misses})] \times 100\ \%$$

IV. RESULT AND DISCUSSION

To test the result we have used CloudSim Simulator 1.3.1. When we compared LFU, LRU performance using this simulator we observed the page hit ratio for of LFU is 27.12% is the best compared with LRU with 14.25%, FIFO with 10.15%

Table 4.1

Algorithm	Page Hits
LRU	14.25
LFU	27.12
FIFO	10.15

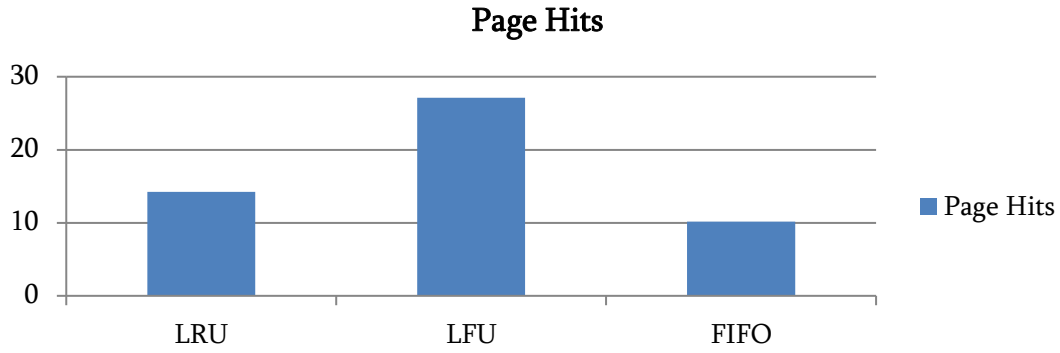


Fig 1: Page Hit % vs Cache Size

For cache size 900, the page hit ratio of LFU is again best compared with LRU and FIFO.
LFU has shown page hits 42.10% which is high when compared LRU with 16.75% and FIFO with 12.21%.

Algorithm	Page Hits
LRU	50%
FIFO	30%
CachoHit	65%

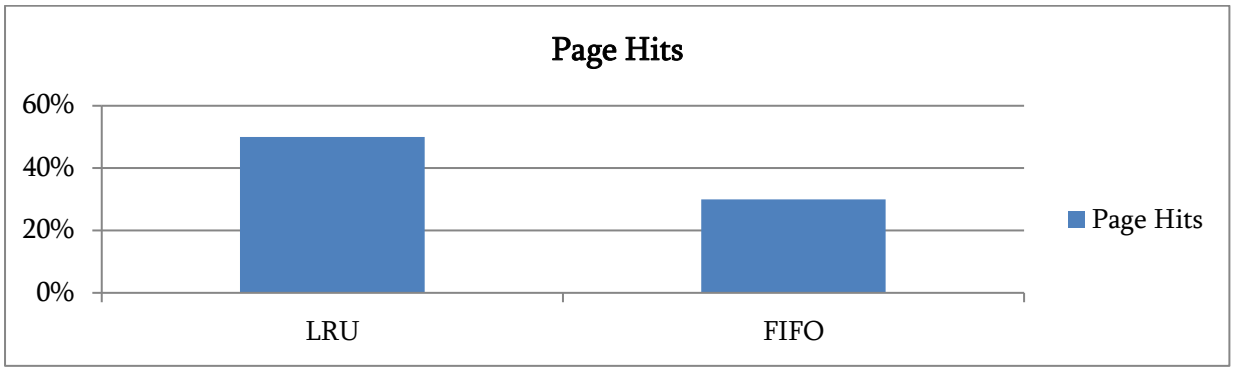


Fig 2: Page Hit % vs Cache Size

From the above results it is observed that page hits of LFU are better than LRU and FIFO in all cases.

V. CONCLUSION

There are various web caching policies have been proposed by different researchers, still lots of overheads and are difficult needs to implement. In this paper, a new replacement policy is developed in order to overcome some of the problems found in the literature. The proposed strategy was able to evict the object with small frequency, size and oldest web object in cache. This was seen in the simulation results through calculating the hit ratio and Byte Hit Ratio. The simulation results showed that proposed.

VI. REFERENCES

[1]. Pushpa C N, Thriveni J, Venugopal K R, Patnaik L M, 2012, Improving Response Time and Throughput of Search Engine with Web Caching, International Journal Of Engineering Research & Technology (Ijert) Volume 01, Issue 04.

[2]. Jhonny Mertz, Ingrid Nunes, "Understanding Application-level Caching in Web Applications: a Comprehensive Introduction and Survey of State-of-the-art Approaches", ACM Computing Surveys, Vol. 9, No. 4, Article 39. Publication date: March 2017.

[3]. V. Sathiyamoorth, Dr. Suresh P, N. Jayapandia, P, Kanmani, Priya, Deva, "An Intelligent Web Caching System for Improving the Performance of a Web- Based Information Retrieval System", International journal on Semantic Web and information systems. Pp. 26-44, 2020.

[4]. A. Mohammed, Osama & Talab, Samani, "Novel Web Cache Replacement Algorithm", International Journal of Computer Science and Information Technology & Security (IJCSITS), ISSN: 2249-9555 Vol.8, No.3, 2018.

[5]. You Fan, Liu Tong, Peng Xiaoyu, Liang Jiahao, Zhang Baili & Zhou Yinian, "An Efficient Web Caching Replacement Algorithm", Artificial Intelligence and Security, pp.479-488, 2020.

[6]. Anil Dudhe, Dr. S.S. Sherekar, Dr. V.M.Thakare, "An Efficient Architecture For Mobile Web Services In Cloud Environment", Journal Of Research And Development, Vol.-10, Issue 13 (Special Issue) : New Trends in Research & Innovation Technology, 2020.

[7]. C. Kumar, S. Marston, "Accelerating The Internet In The Presence Of Big Data: Reducing User Delays By Leveraging Historical User Request Patterns For Web Caching", Journal of Information Systems and Technology Management, Published by TECSI FEA USP, Brazil, Vol. 16, 2019.

[8]. S. M. Shahtouri, M. Rezazad and R. T. B. Ma, "Frozen Cache: Mitigating Filter Effect and Redundancy for Network of Caches," in IEEE Access, vol. 9, pp. 139725-139738, 2021.

[9]. V. A. Kakde , S. K. Mishra, A. Sinhal ,M. S. Mahalle, "Survey of Effective Web Cache Algorithm", International Journal of Science and Engineering Investigations (ijsei), Volume 1, Issue 1, 2012.

[10]. A. Ioannou and S. Weber, "A survey of caching policies and forwarding mechanisms in information-centric networking," IEEE Communications Surveys & Tutorials, vol. 18, no. 4, pp. 2847–2886, 2016.