Network Cache System Simulation

Shubham Suresh Chandak
Computer & Information Science & Engineering
University of Florida
Gainesville, FL - United States
shubham.chandak@ufl.edu

Vinit Bharatkumar Jain
Computer & Information Science & Engineering
University of Florida
Gainesville, FL - United States
vinitjain@ufl.edu

***Abstract*— To analyze the performance of different cache replacement policies used by institutional networks we present a probabilistic event-driven network cache simulator. This paper presents the analysis of LRU and FIFO cache implementation along with their effectiveness for different scenarios. This system can help institutions analyze their network use case and make a cost-effective decision in choosing the right cache implementation without spending much on infrastructure and resources. However, to derive a reasonable conclusion, probability distributions of different events are assumed with suitable parameters to simulate a real network cache system.**

Keywords— FIFO, LRU, Cache replacement policies, Probability distributions

Introduction

For a large institutional network, it is necessary to reduce latency and improve the response time given the large frequency of outgoing requests from the network. Increasing the access link bandwidth can be a possible solution but it is not economical. To address this an institutional cache system can be set up to lower the pressure on the access link and it can also significantly improve the average response time of requests. It is thus necessary to configure the cache system with an optimal cache replacement policy. In this paper, we analyze the two most effective and commonly used cache replacement policies – LRU (least recently used) and FIFO(first-in-first-out). We first describe the features of our simulation system followed by the assumptions on the probability distributions of various events in the simulation. Then we show the analysis of different cache replacement policies with different choices of parameters and finally we conclude with the result of our analysis.

I. SIMULATION

A. Assumptions

- There is only one origin server where there are N files/web-pages residing far away in the internet.
- There is only one user requesting files from the origin server.
- The user makes file requests according to a Poisson process with rate λ requests per second. Each request is for file i with popularity/probability pi.
- File i has a size Si , which is a sample drawn from a Pareto distribution (heavy tail), FS, with mean μ (μ = 1 MB).
- For i = 1, . . . , N, independently drawn from another Pareto distribution, where pi=qi/ **Σ** qj
- There is a cache server in the institution network. A user request goes to the cache first. If the cache contains the requested file, it transmits the file to the user. Else the request goes to the origin server on the Internet.
- In the in-bound direction at the access link, there is a first-in-first-out (FIFO) queue with infinite capacity. The returned files enter the FIFO queue and are transmitted by the access link in order.
- After that the file goes to the cache first and is cached, and then goes to the user.
- The propagation time within the institution network is zero.
- The round-trip (propagation) time between the institution network and the origin server is D, which is a random variable with a lognormal distribution
- After file i is completely transmitted from the access link, it is immediately cached at the cache server (which may involve replacing some cached files). After an additional time, Si/Rc, it is received by the user.

B. Event Driven Model

The simulation was developed in python and follows object-oriented design principles. It is an event driven model which generates events with their respective event time, sent to the queue and are executed in order of their arrival by advancing the simulation-clock to the respective event time. Below are events which are generated and processed:

new-request-event*:* This event corresponds to a new user request for a file/resource.
file-received-event: This event represents that a file has been received by the user.
arrive-at-queue-event*:* This event corresponds to a file arriving at the in-bound FIFO queue.
depart-queue-event*:* This event represents that the access link has finished the transmission of a file.

C. Parameter Selection

The simulation can be performed with different values of parameters, however, to simulate like a real cache system analysis is performed with below settings of parameters.

- Total requests = 10000
- Number of files = 10000
- λ = 50 requests/sec
- Mean for lognormal distribution for roundtrip = 500ms
- Standard deviation for lognormal distribution for roundtrip = 400ms
- Cache size = 1000
- Access link bandwidth = 15Mbps
- Institutional link bandwidth = 100Mbps
- Pareto distribution parameter = 2 (heavy tail)

II. Cache Replacement Policy

The cache replacement policy refers to the rule that define the strategy to evict the item from the cache when the cache-capacity is reached, and new item needs to be inserted in it. There are various cache-replacement policies out there but the most effective and commonly used are LRU and FIFO, and we have analyzed the performance of these two with our simulation model in this paper.

A. *LRU*

Given the capacity of cache (number of files that cache can hold at a time), the LRU caching scheme is to remove the least recently used file when the cache is full and a new file is referenced which is not there in cache, the new file is then stored in the cache. In our simulation, Ordered Dictionary is used for implementing LRU cache.

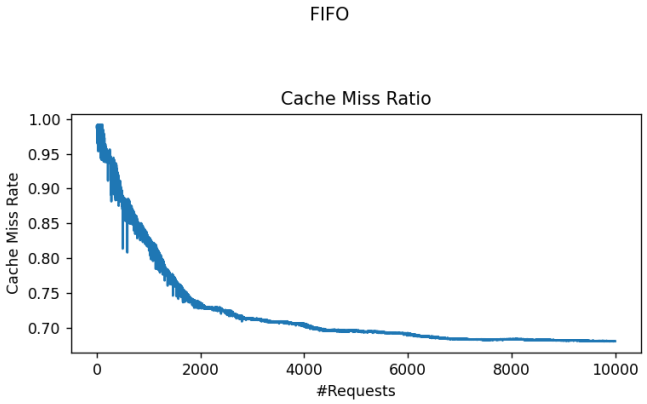
B. *FIFO*

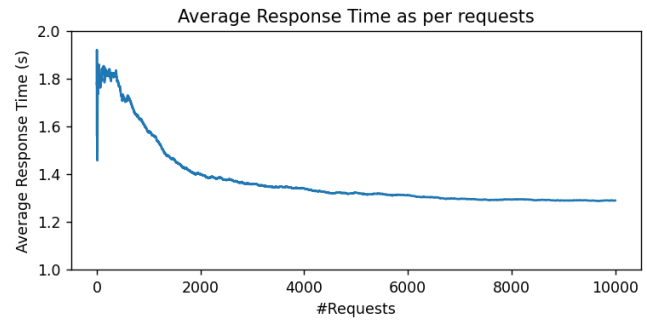
A First-In-First-Out cache evicts the file in the order they were added when the capacity of cache is reached, without any regard to how often or how many times they were accessed before. In our simulation, Python Dictionary and List are used for implementing FIFO cache.

III. Result and Analysis

After performing simulation as per the parameters mentioned in section [1][C], below is the performance evaluation for FIFO and LRU cache replacement policies in terms of cache miss ratio and average response time of requests.

A. *FIFO Analysis*

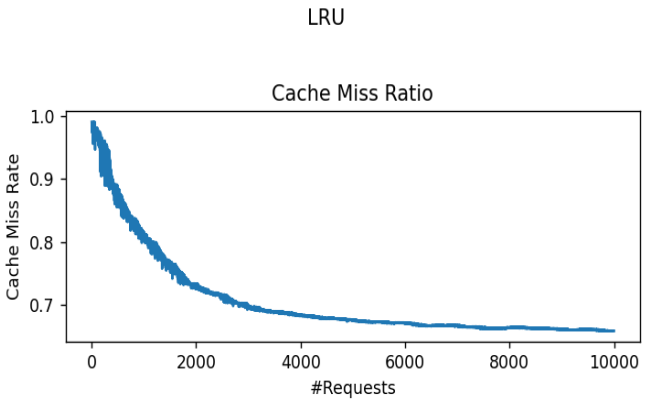


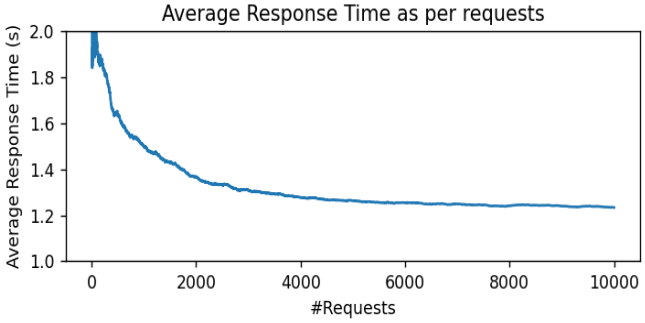


As we can see from the above plots, for simulation on ten thousand requests, we get an average response time of 1290ms, and the cache miss ratio of 68.06%. We can also see that the cache miss ratio declines initially and then the value converges to ~68%. Similarly, the average response time also declines initially and converges near ~1300ms. This is expected as with subsequent requests the files with higher probability/popularity are likely to exist in the cache that lowers the response time, thus reducing the overall average response time.

B. *LRU Analysis*

Similarly, for LRU we get an average response time of 1235ms, and the cache miss ratio of 65.88% for simulation over ten thousand requests. The graph follows a similar trend as FIFO replacement policy which is expected in general for any cache policy. The average response time converges near ~1200ms and cache miss ratio converges near ~67% which is slightly better in comparison to FIFO.





IV. Conclusion

As per our analysis of simulation over ten thousand requests for FIFO and LRU cache replacement policies, we can say that the average response time can be significantly reduced by any of the two mentioned cache policies. There is more than 30% reduction on average response time for both FIFO and LRU cache, however, LRU performs slightly better than FIFO. This conclusion is specific to the constraints on which the simulation was performed and can vary based on the selection of different parameters for simulation. We can say that if the distribution of data can be derived from the institutional network traffic, then performing such simulation for an institutional network can help in deciding the optimal cache replacement policy which can significantly lower the burden on access link and reduce the latency for network users.