Web Cache Simulation

NishaNataraj,Mrs. S. Ajitha,Mr.Jagannatha
M. S. Ramaiah Institute of Technology, Bangalore

Abstract

Web caching is the storage of Web objects near the user to allow fast access, thus improving the user experience of the Web surfer. Examples of some Web objects are Web pages (the HTML itself), images in Web pages, etc. We assume that system has only one server who serves the clients web requests. The arrival of requests is uncertain and cannot be predicted.
Simulation is the most commonly usedmethod to explore new proposals due to both itsflexibility and the relatively reduced time taken to obtain performance results. This paper presents a simulation system to simulate web proxy cache systems. Experimental Results show that the maximum requests's waiting time is noted to be between 0 and 62 seconds. Therefore, the requests waiting for 2 minutes is bearable and does not require an additional server.Similarly, the server's service time for most number of requests is between 3 and 4 minutes.

1. Introduction

Web Cache Simulation system is a system where web requests arrive. The server serves the requests. The server spends some time for each request and that time is referred to as service time. The service time may differ from request to request. In web server the number of arrival of requests is more during the day, during that time server may be busy. During other time server may be idle. Server spends some time for each of requests which is varying. Requests are served in the order of their arrival i.e. FIFO (First In First Order).

Web caching has the following advantages:

- Faster delivery of Web objects to the end user.
- Reduces bandwidth needs and cost. It benefits the user, the service provider and the website owner.
- Reduces load on the website servers.

The remainder of this paper is organized as follows: Section 2 discusses some related work of web caching simulation environments. Section 3 illustrates the methodology followed for the simulation which includes the flow chart and algorithm. The corresponding results and discussions for it is dicussed in section 4

2. Related work

The main characteristics of arepresentative set of cache and network simulators is presented in Cao [4] proposed the Web Cache simulator which is a fast trace-driven.Thesimulator manages a priority queue to implementreplacement algorithms. This simulator focuses mainlyon caching techniques and, to our knowledge nocomparison against a real system has been done.Davison [7] developed the trace-driven Network andCache Simulator (NCS) which simulates caching as anoptional feature. Experiments [8] focuses in how tovalidate portions of networks models considered in thesimulator. The process compares responses times in thetraces against the simulated latencies.

Proxim[3] is a caching and network simulator thatemulates proxy caches using log traces. Estimations of
mean latencies to validate Proxim were compared [9]against measured latencies (i.e., logs). No performance
comparison was done for caching characteristics,because experiments use an infinite cache size.Network Simulator (NS) [13] includes a simple HTTPclass to model a simple cache of infinite size that does
neither implement replacement algorithms, nor measurescaching performance. Extensive verification tests [11]have been performed with this tool.

MSE [5] is a trace-driven proxy cache simulator. This simulator models cache management techniques using classes. The tool is composed by two mainmodules, a filter module and the cache simulator. Thefilter module prepares original proxy logs to feed thecache module. In addition, the filter program estimateslatency related parameters per request used in the cachesimulator. These parameters estimates a mean responsetime per request (conformed by a latency per eachrequest and a penalty time for each miss).

The cache simulator module simulates a web cachestructure. This module takes as input a log prepared by
the filter module, a parameters file with the cachecharacteristics, and it generates a statistics file. Mainperformance statistics obtained are: hit ratio (HR), byte
hit ratio (BHR) and the mean response time per request.

3. Methodology

In this system, queuing problem may arise due to uncertainty of arrival of requests. Some time more requests may arrive and some other time the arrival rate of request may be less. When number of request arrival is more, the waiting time of requests may be more since the single server spends some time in processing for each request. The arrival rateof request is not predictable and thus the necessity of simulation arises. It all depends on server serving the requests and time taken by it to serve.

Inputs such as inter arrival time, service time are provided through random numbers which are generated programmatically. The lookup table is constructed and the generated numbers are passed to it, to get suitable inter arrival time or service time. These serve as input values to the problem.

The generation of random numbers ensures that the values of input are not repeated. These provide numbers of different ranges which helps to formulate the problem in an efficient manner.

From the problem, some of the expected output parameters are request waiting time, time at which service begins, time at which service ends and idle time of server. The values of these parameters help us to simulate and analyze the problem. Parameters like request waiting time, idle time helps us to decide whether to increase the number of servers or continue with the same.

For example, if average waiting time of request is more, then an addition of server can be made to ease the load from the existing server and increase the request satisfaction by cutting down on the server turnaround time

In this system, we analyze average waiting time of request, average service time of server and idle time of server. These parameters help us to decide if additional servers are required for processing the load. This is done by taking different trials for fixed numbers of requests. After each trial, the parameter values are noted accordingly.

This problem is simulated by calculating the inter arrival time of requests, average waiting time of requests, average response time of the server and idle time of the server. Inter arrival time and service time is randomly distributed. In order to compute summary statistics following procedure is followed.

For the first request inter arrival time and arrival time will be assigned to zero. For the second request inter arrival time is generated and calculate arrival time using following formula.

Arrival Time of request(i) = Arrival time of request(i-1) + inter arrival time of request(i) where i=1,2..n

If the server is idle then service begin time will be same arrival time and waiting time of the request will be zero. Service end will be calculated as follows.

Service End of request(i) = Service begin time of request(i) + service time of request(i);

wherei=1,2..n

Finally average waiting time and average service timeof requests and idle time of server has to be calculated using following formulas.

i) Average waiting time

Average waiting time =

Total time requests wait in queue
Total number of requests

ii) Average Service time

Average Service time =

Total service time (minutes)
Total number of requests

iii) Probability of Idle server

Probability of Idle server =

Total idle time of server(minutes) Total runtime of simulation

3.1 Flow Chart

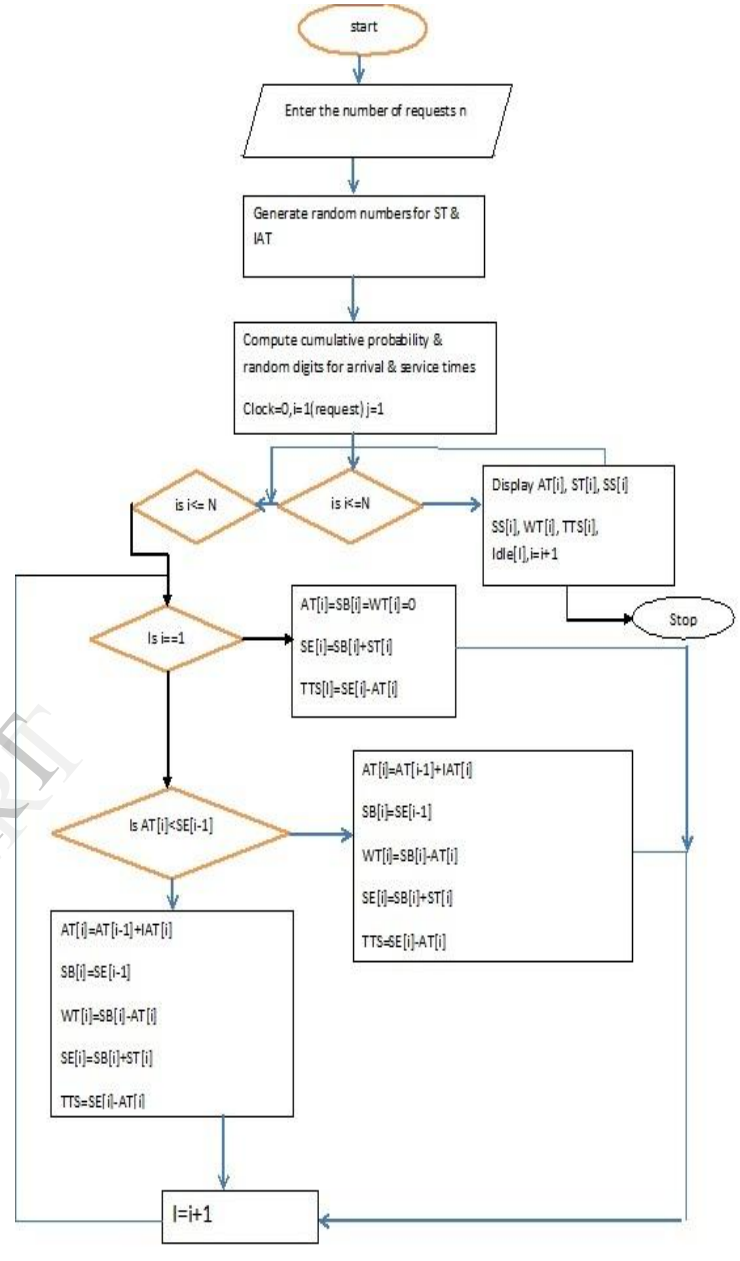

Figure 1: flow chart

3.2 Algorithm

Begin

{

1. Start
2. Enter the number of requests n
3. Generate the random numbers for service time and inter arrival time using uniform distribution
4. Calculate the cumulative probability for service time and inter arrival time
5. If i is 1 then Initialize arrival time, service beginning time and waiting time to 0
   at[i]=0,sb[i]=0,wt[i]=0
6. Compute service ending time using the formula *send[i]=sbegin[i]+st[i]*
7. Compute average waiting time, average service time ,total time in system for each entry

tt[i]=st[i]+wt[i];

awt=awt+wt[i];

ast=ast+st[i];

itt=itt+it[i];

atts=atts+tt[i];

8. Compute the total average waiting time,average service time and the idle time and the average total time spent in system
   awt=awt/n;

ast=ast/n;

itt=itt/send[i];

atts=atts/n;

9. Stop

}

end

4. Results and Discussion

In table 1 we have given the results obtained for different requests versus average response time and a graph is presented in the figure. The graph plots average response time of client web requests to number of requests.

Table 1:average response time

No of requests	Average response time (in seconds)
50	4.6
150	4.45
250	4.5
400	4.47
550	4.4
700	4.39
750	4.34
850	4.45

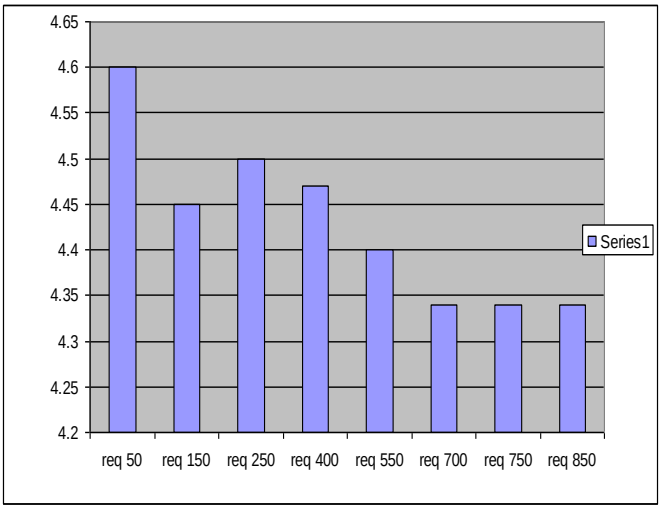

Figure 2: graph for response time

In table 2 we have given the results obtained for different requests versus the average waiting time and a graph is presented in the figure.This graph plots average waiting time of requests to number of requests.

Table 2:average waiting time

No of requests	Average waiting time (in seconds)
50	7.28
150	45.73
250	55.17
400	64.34
550	66.12
700	62.35
750	59.75
850	61.89

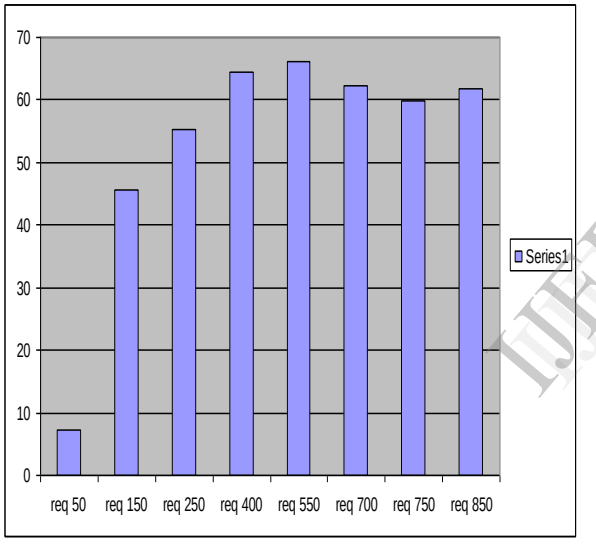

Figure 3: graph for waiting time

In table 3 we have given the results obtained for different requests versus the average idle time of the server and a graph is presented in the figure.This graph plots average idle time of server which doesnot serve any request at some point of time

Table 3:average idle time

No of requests	Average idle time of server (in seconds)
50	0.04
150	0.01
250	0.01
400	0.006
550	0.004
700	0.004
750	0.004
850	0.004

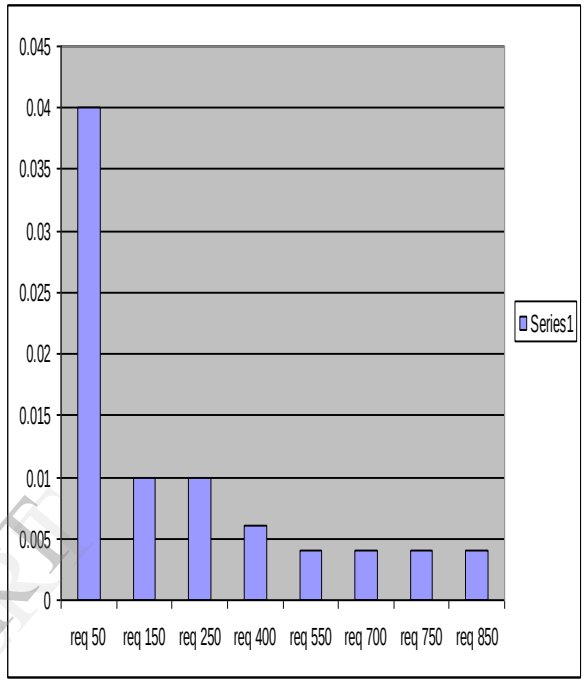

Figure 4: graph for idle time

5. Conclusion

The simulation of "Web Cache system" has provided the values for parameters such as average waiting time, average response time of server etc.

The maximum requests's waiting time is noted to be between 0 and 62 seconds. Therefore, the requests waiting for 2 minutes is bearable and does not require an additional server.

Similarly, the server's response time for most number of requests is between 4 and 5 secondss.

By observing and analyzing the various results obtained, it can be concluded that this system can be managed with only a single proxy server. Necessity of the second server does not exist.

6. References

[1]http://pages.cs.wisc.edu/~cao/webcache-simulator.html

[2]http://www.gii.upv.es/web_architecture/tools/paper-20061031125802-apont.pdf
[3] R. Caceres, F. Douglis, A. Feldmann, G. Glass, and M.Rabinovich, "Web Proxy caching: The Devil is in the Details", *SIGMETRICS Performance Evaluation Review*, 26,1998.
[4] P. Cao, Wisconsin Web Cache Simulator, http://www.cs.wisc.edu/~cao/, 1997.
[5] L.G. Cárdenas, J. Sahuquillo, A. Pont, and J.A. Gil, "TheMultikey Web Cache Simulator: a Platform for DesigningProxy Cache Management Techniques. *Proceedings of the12th Euromicro Conference on Parallel, Distributed andNetwork based Processing*, A Coruña, Spain, 2004.
[6]http://www.cs.utexas.edu/~dahlin/Classes/WebOS/hw/cache-simulator.html
[7] B.D. Davison, "*NCS: Network and Cache Simulator – AnIntroduction",* (Tech. Rep. DCS-TR-444), New Jersey, USA, Rutgers University, Department of Computer Science,2001.
[8] B.D. Davison, "HTTP Simulator Validation Using RealMeasurements: A Case Study", *Proceedings of the NinthIEEE International Symposium on Modeling, Analysis andSimulation on Computer and Telecommunication Systems*,Cincinnati, USA, 2001.
[9] A. Feldmann, R. Cáceres, F. Douglis, G. Glass, and M. Rabinovich, "Performance of Web Proxy Caching in Heterogeneous Bandwidth Environments", *Proceedings of the 18th Conference of the IEEE Computer and Communications Society*, New York, USA, 1999.
[10] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter,P. Leach, and T. Berners-Lee, "Hypertext Transfer Protocol–HTTP/1.1, RFC 2616", Network Working Group, 1999.
[11] S. Floyd, "Validation Experiences with the NS Simulator",*Proceedings of the DARPA/NIST - - Network SimulationValidation Workshop*, Fairfax, USA, 1999.
[12]GNU-wget, http://www.gnu.org/software/wget/wget.html.
[13] Network Simulator, http://www.isi.edu/nsnam/ns.