Effects of the coherency on the Performance of the Web Cache Proxy Server

Angel Vassilev Nikolov,

National University of Lesotho

Summary
We develop an analytical model of the web proxy cache and obtain the hit ratio in terms of the incoming requests rate, replacement distribution and document's modification rate. We choose a queuing network to evaluate the decrease of the response time. Numerical results show that securing a perfect coherency leads to performance degradation which could be significant in extreme cases.

Key words:
web cache coherence protocols, hit ratio, queuing network

1. Introduction

The wide adoption of Internet has fundamentally altered the ways in which people communicate and conduct businesses. As the number of web sites grows exponentially a lot of latencies and bandwidth consumption are experienced. Many efforts have been done to make web page delivery to the user faster, to improve servers' performance, and to reduce network traffic. In order to achieve these goals server providers utilize caching whereby frequently used files are copied and stored "near" to user on the network [13]. The proxy server's role is to accept requests from the users and forward them on their behalf. If the proxy cache does not contain the document it makes a request to the remote server, which replies to the proxy. The proxy sends the file to the requestor, and stores it locally in a cache. Any further requests for the same document will be satisfied from the cache without contacting the origin server, thus reducing its load and relieving the networks. Caches can be arranged in hierarchical, distributed or hybrid schemes to satisfy bigger client community [5, 6, 14].
The effects of caching systems, however, are two-fold. As time progresses some pages are being modified at the remote site, and if not properly updated at the proxy server, there would be a possibility that users download stale (old) copies. Cache coherency protocols are designed to eliminate or reduce this adverse effect [1, 12]. Prefetching the most popular web pages also contributes to this goal [9, 10, 12, 13]. Since extra message exchanges and downloads are required to secure the user reads the most recent copy, these protocols pose burden to the network and increase the response time. The need to understand and evaluate their effects on the overall performance is vital. Trace-driven simulation is mostly used as a tool for the purpose of evaluating different performance factors, such as hit ratio, latency, mean response time, error rate, queuing of requests, connection length, etc. [1, 4, 5, 7, 8]. Comparison of well designed and detailed simulator against a real system shows a deviation of several percents [5]. Development of comprehensive, flexible, and very much detailed simulators, however, could be very costly and time consuming. Analytical models based on queuing theory are simple and easy to use [2, 3, 5, 11]. Validation of the analytical models against event-driven simulation shows that they can be useful in the early design cycle because approximation is satisfactory. None of the above mentioned queuing models, however, deals with the specific issues of the cache coherence. The aim of our work is to fill this gap.

2. Definition of the model

The benefit of current web caching schemes is limited by the fact that only a fraction of web data is cacheable. Personalized data, authenticated data, dynamically generated contents are not cacheable. Some researchers estimate the ratio of non-cacheable document between 37.5% and 53% [9].

Web cache coherency falls into two major classes: strong coherency and weak coherency [1, 12]. Strongly coherent protocols provide perfect consistency, that is each read request returns the most recent copy. Strong coherence is implemented by two protocols-client validation and Invalidate protocols. Client validation protocol uses a conditional HTTP directive-*'Get-if-modified'*. Each time the client requests the a document, and the document is cached, a message is sent to the remote Web server, which returns a *' Not modified'* message if the document has not been modified or alternatively the whole modified document. Invalidation protocol requires each server to maintain a list of clients that have requested a particular document. Each time the document is changed, an Invalidate message is sent to all clients in the list. If the cached document has been

Manuscript received April 5, 2009
Manuscript revised April 20, 2009

invalidated the next request will not be satisfied from the cache but from the remote server. Weak coherence web caching is mostly implemented by writing a *'Time-to- live'* value in the '*expires'* header fields. Obviously this protocol allows for the possibility of retrieving an out-of-date copies.

Next we will derive expression for the hit ratio when a strong cache coherence is adopted. According to the nature of the caching system a document can be in one of the following states: cached (*c*), cached and modified (*cm*), and uncached (*u*). We assume that both request arrivals and modification arrivals follow a Poisson distribution with parameters λ and λ_m, respectively, and evictions (replacements) have a general distribution. In the case of Invalidation protocol λ_m is actually the rate of the invalidation messages. Transitions between states are illustrated in Fig. 1.

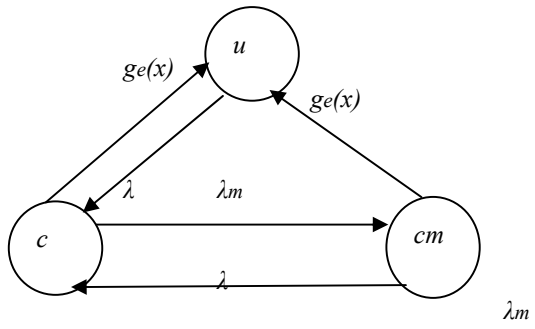

Fig.1

We introduce the following notations

Pc(x) Probability [in the equilibrium state document is in state *c* and the elapsed time since joining this state lies between *x* and *x+dx*]

Pcm(x) Probability [in the equilibrium state document is in state *cm* and the elapsed time since joining this state lies between *x* and *x+dx*]

Pz Probability [in the equilibrium state document is in state *z*; *z=c, cm or u*]

Fe(x) cumulative distribution function (c.d. f.) of the eviction time

fe(x) probability density function (p. d. f.) of the eviction time

$$\eta = \int_0^\infty x f_e(x)dx$$

$\underline{f_e(s)}$ Laplace transform of $f_e(x)$

$g_e(x) = \dfrac{f_e(x)}{1-F_e(x)}$ rate of evictions (replacements)

We set the following integro-differential equations

$$\left(\frac{d}{dx} + g_e(x) + \lambda_m\right)P_c(x) = \lambda P_{cm}(x) \quad (1)$$

$$\left(\frac{d}{dx} + g_e(x) + \lambda\right)P_{cm}(x) = \lambda_m P_c(x) \quad (2)$$

$$P_u\lambda = \int_0^\infty P_c(x)g_e(x)dx + \int_0^\infty P_{cm}(x)g_e(x)dx \quad (3)$$

having the following boundary and initial conditions

$$P_c(0) = \lambda P_u \quad (4)$$

$$P_{cm} = 0 \quad (5)$$

$$P_c + P_{cm} + P_u = 1 \quad (6).$$

Let $u_c(x) = \dfrac{P_c(x)}{1-F_e(x)}$ and $u_{cm}(x) = \dfrac{P_{cm}(x)}{1-F_e(x)}$.

Equations (1) and (2) now can be rewritten as follows

$$\left(\frac{d}{dx} + \lambda_m\right)u_c(x) = \lambda u_{cm}(x) \quad (7)$$

$$\left(\frac{d}{dx} + \lambda\right)u_{cm}(x) = \lambda_m u_c(x) \quad (8).$$

Equations (7) and (8) can be solved using Laplace transform:

$$u_c(x) = u_c(0)\left(\frac{\lambda}{\lambda+\lambda_m} + \frac{\lambda_m e^{-(\lambda+\lambda_m)x}}{\lambda+\lambda_m}\right) \quad (9)$$

$$u_{cm}(x) = \frac{u_c(0)\lambda_m(1-e^{-(\lambda+\lambda_m)x})}{\lambda+\lambda_m} \quad (10).$$

$P_c(x)$ and $P_{cm}(x)$ can now be expressed:

$$P_c(x) = u_c(0)(1-F_e(x)\left(\frac{\lambda}{\lambda+\lambda_m} + \frac{\lambda_m e^{-(\lambda+\lambda_m)x}}{\lambda+\lambda_m}\right) \quad (11)$$

$$P_{cm}(x) = \frac{u_c(0)(1-F_e(x))\lambda_m(1-e^{-(\lambda+\lambda_m)x})}{\lambda+\lambda_m} \quad (12).$$

From (3), (4), (5), (6), (11) and (12) after some transformations the steady-state probabilities can be determined:

$$P_c = \frac{\lambda}{1+\lambda\eta}\left(\frac{\lambda\eta}{\lambda+\lambda_m} + \frac{\lambda_m}{(\lambda+\lambda_m)^2} - \frac{\lambda_m f_e(\lambda+\lambda_m)}{(\lambda+\lambda_m)^2}\right) \quad (13)$$

$$P_{cm} = \frac{\lambda}{1+\lambda\eta}\left(\frac{\lambda_m\eta}{\lambda+\lambda_m} - \frac{\lambda_m}{(\lambda+\lambda_m)^2} + \frac{\lambda_m f_e(\lambda+\lambda_m)}{(\lambda+\lambda_m)^2}\right) \quad (14)$$

$$P_u = \frac{1}{1+\lambda\eta} \quad (15).$$

If no coherence is implemented ($\lambda_m = 0$) the steady-state probabilities are

$$P_c = \frac{\lambda\eta}{1+\lambda\eta} \quad (16)$$

$$P_u = \frac{1}{1+\lambda\eta} \quad (17).$$

Proof of (16) and (17) is very similar to that of (13)-(15) and is therefore omitted.

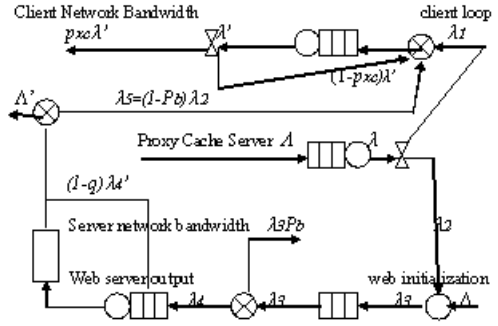

Fig. 2 redrawn from [2]

The operations involved in the delivery of a requested document can be represented by Jackson network (Fig. 2). The network has four queues (nodes) and two delay centers (network bandwidths). If the document can be retrieved from the cache the request is directed to the client network ($\lambda_1 = \lambda * hit$), otherwise it goes to remote server ($\lambda_2 = \lambda * miss$).

At the servers the requests undergo a process of initialization, which take times *Is* and *Ixc* respectively, then processed and passed on to the networks. If the actual file size exceeds the output buffer of the servers, it returns for further processing and transmission with probabilities 1-*pxc* and 1-*q* respectively, where $pxc = \frac{F}{B_{xc}}$ and $q = \frac{F}{B_s}$ (F-average file size I n bytes, *Bxc* output buffer of the proxy server and *Bs* –output buffer of the remote server in bytes). Other notations: *Λ* arrival rate of external requests, *Yxc* and *Ys* static server time for proxy and remote servers, *Rxc* and *Rs* dynamic server rate for proxy and remote servers, K-buffer size of the remote server(in requests), *Nxc* proxy server bandwidth, and *Ns* remote server bandwidth.

The overall response time *Txc* (overall sojourn time) of a request in a Jackson network can be expresses as a sum of the sojourn times in each node. In [2] the following expression for *Txc* is presented:

$$T_{xc} = \frac{1}{\frac{1}{I_{xc}} - \lambda} + hit\left(\frac{\frac{F}{B_{xc}}}{\frac{1}{Y_{xc} + \frac{B_{xc}}{q_{xc}}} - \frac{\lambda}{q_{xc}}} + \frac{F}{N_c}\right) + miss\left(\frac{1}{\frac{1}{I_s} - \lambda_3} + \frac{\frac{F}{B_s}}{\frac{1}{Y_s + \frac{B_s}{R_s}} - \frac{\lambda_4}{q}} + \frac{F}{N_s}\right)$$

3. Numerical Example

We assume that the ratio of cacheable documents is 0.5, so that the hit ratio *hit =0.5Pc,* and the miss ratio *miss*=1-*hit=1-0.5Pc.* Time to eviction follows erlangian distribution with shape *k=4,* and rate λe=10 $\left[1/\text{sec}\right]$, λ=100 $\left[1/\text{sec}\right]$ and λ_m varies.

The other parameters in our example are: Λ=100 $\left[1/\text{sec}\right]$, *Is=Ixc=0.004* sec, *Bs=Bxc=2Kbytes, Ys=Yxc=0.000016sec, Rs=Rxc=1250Mbytes/s, Ns=1Mbit/s, Nc=100Mbit/s, F=50Kbytes* and *K=100.* Needless to say it is worthwhile to retrieve most of the requests from the web cache because of the higher bandwidth.

λm	hit	miss	Txc
0	0.48780487804878	0.51219512195122	0.0386632887613242
1	0.483094666914369	0.516905333085631	0.0389738290266594
2	0.47847449191649	0.52152550808351	0.0392791147747495
3	0.47394179758741	0.52605820241259	0.0395792852805813
4	0.469494124196608	0.530505875803392	0.039874474791044
5	0.465129103312126	0.534870896687874	0.0401648127534972
6	0.460844453606431	0.539155546393569	0.0404504240318781
7	0.456637976891217	0.543362023108783	0.0407314291111381
8	0.452507554366702	0.547492445633298	0.0410079442907407
9	0.448451143072039	0.551548856927961	0.0412800818678983
10	0.444466772524344	0.555533227475656	0.0415479503111831
11	0.440552541534798	0.559447458465202	0.0418116544250967
12	0.436706615191001	0.563293384808999	0.0420712955061467
13	0.432927221995583	0.567072778004417	0.0423269714909399
14	0.429212651151683	0.570787348848317	0.042578777096766
15	0.425561249986612	0.574438750013388	0.0428268039551147
16	0.421971421505546	0.578028578494454	0.0430711407385396
17	0.418441622067667	0.581558377932333	0.0433118732812539
18	0.414970359177677	0.585029640822323	0.0435490846938179
19	0.411556189386036	0.588443810613964	0.0437828554722561
20	0.408197716291759	0.591802283708241	0.0440132636019172
21	0.404893588641965	0.595106411358035	0.0442403846563733
22	0.401642498522755	0.598357501477245	0.0444642918916318
23	0.398443179636351	0.601556820363649	0.0446850563359217
24	0.395294405659732	0.604705594340268	0.0449027468752932
25	0.392194988680301	0.607805011319699	0.0451174303352599
26	0.389143777704421	0.610856222295579	0.0453291715586958
27	0.38613965723486	0.61386034276514	0.0455380334801876
28	0.383181545913497	0.616818454086503	0.0457440771970299
29	0.380268395225793	0.619731604774207	0.0459473620370395
30	0.377399188263792	0.622600811736208	0.0461479456233563
31	0.374572938544593	0.625427061455407	0.0463458839363848
32	0.371788688881396	0.628211311118604	0.0465412313730251
33	0.369045510304433	0.630954489695567	0.0467340408033302
34	0.366342501029227	0.633657498970773	0.046924363624721
35	0.363678785469765	0.636321214530235	0.0471122498138814
36	0.361053513294342	0.638946486705658	0.0472977479764489
37	0.358465858521915	0.641534141478085	0.0474809053946102
38	0.355915018656979	0.644084981343022	0.0476617680727052
39	0.353400213861041	0.646599786138959	0.0478403807809361
40	0.350920686158921	0.649079313841079	0.0480167870972733

Table1. *hit, miss,* and response time *Txc* as a function of λm.

Results are summarized in Table 1. Both hit ratio and response time decrease as the modifications at the remote site become more intensive. For λm= 40 the hit ratio is down by (0.48780487804878-0.350920686158921)/ 0.48780487804878*100=28%, and the response time increased by (0.0480167870972733-0.0386632887613242)/0.0480167870972733=24%, so the effect of introducing perfect coherency can result in a significantly lower performance.

4. Concluding Remarks

The major contribution of this paper is an analytical model which allows us to evaluate the impact of the coherence on the user perceived performance of the proxy server. Degradation of the performance can be significant and should not be neglected. Trade-off should be found between the targeted degree of coherence and the amount of the caused latency.

The obtained expressions could be included in a spreadsheet and used at the early design stage.

References

[1] A.Belloum, B. Hertzberger; Maintaining web cache coherency, Information Research, vol, 6, October 200, No. 1

[2] T. Berczes, et.al., Analyzing web server Performance Models with the Probabilistic Model Checker PRISM, RISC-Linz Report Series No. 09-17, 2007

[3] I. Bose, H.K.Cheng; Performance Models of a firms Proxy Cache Server; Decision Support Systems and Electronic Commerce, 2000, No. 29, pp. 45-57

[4] M. Busari; Comparison of Cache Replacement Algorithms in Web proxies; http://www.cs.usask.ca/faculty/carey/papers/Muda 855.doc

[5] H.Che, Y.Tung, Z. Wang; Hierarchical Web Caching Systems: Modeling, Design And Experimental Results; IEEE Journal on Selected Areas in Communication, vol,20, 2002, No.7, pp.1305-1314

[6] D. M. Dias et.al., US Patent 64906/5-Scalable Cache, 2002

[7] P.Du, J. Subhlok; Evaluation of Performance of Cooperative Web Caching with Web Polygraph, 7th International Workshop of Web Content Caching and Distribution, Boulder, CO, July 2002

[8] A. Feldman et. al.; Performance of Web Proxy Caching in Heterogeneous Bandwidth Environments, Proceeding of INFOCOM 99, 1999

[9] E. Jaffe; Web Caching, Seminar on Databases and Internet, Hebrew University, Fall 2002, pp. 1-53

[10] V.N. Padmanabhan, J. C. Mogul; Using Predictive Prefetching to Improve World Wide Web Latency, ACM Computer Communication Review, July 1996

[11] L. P. Slothouber; A Model of Web Server Performance, Telecommunication Systems, vol. 16, No.3, March 2001, pp. 361-378

[12] J. Sustersic, A. Hurson; Coherence Protocols for Bus-based and Scalable Multiprocessors, Internet and Wireless Distributed Computing Environment: A Survey; Advances in Computers, vol. 59. 2003, pp. 211-278

[13] C. Umapath, J. Raja; A Prefetching Algorithm for Improving Web Cache Performance; Journal of Applied Sciences, vol.15, No.6, 2006, pp. 3122-3127

[14] J. Wang, A Survey of Web caching Schemes for the Internet, ACM SIGCOMM Computer Communication Review, vol. 29, issue 5, 1999, pp. 36-46

Angel Vassilev Nikolov received the BEng degree in Electronic and Computer Engineering from the Technical University of Budapest, Hungary in 1974 and the PhD degree in Computer Science from the Bulgarian Academy of Sciences in 1982 where he worked as a Research Associate. In 1989 he was promoted to Associate Research Professor in Bulgaria. Dr Nikolov also served as a Lecturer of Computer Science at the National University of Science and Technology, Bulawayo, Zimbabwe and at the Grande Prairie Regional College, Alberta, Canada and as an Associate Professor at Sharjah College, United Arab Emirates. Currently he works for the National University of Lesotho, Roma, Lesotho. His research interests include computer architecture, performance evaluation of multiprocessors, and reliability modeling.